UNIVERSIDAD CARLOS III DE MADRID

ESCUELA POLITÉCNICA SUPERIOR

Universidad
Carlos III de Madrid

GRADO EN INGENIERÍA INFORMÁTICA

Trabajo de Fin de Grado

ESTUDIO SOBRE ATAQUES DE SQL INJECTION EN LA PLATAFORMA ANDROID

Autor: Luis Julián Arjona Pérez

Tutor: Jorge Blasco Alís

Junio de 2014

Agradecimientos.

Mi primera etapa universitaria no comenzó con muy buen pie, me incorporé a un curso ya empezado al que no supe engancharme, no recuerdo aprobar asignatura alguna, y salí casi huyendo, pensando que la universidad me venía grande.

Cuatro años después, con un ciclo formativo de grado superior con excelentes notas, y dos años de experiencia como administrador de sistemas, volvía a la universidad con fuerzas renovadas, y sobre todo, con mucha confianza en mí.
¡¡ Esta vez la universidad no me vendría grande, esta vez no podría conmigo!!

Ahora, en este momento, puedo echar la vista atrás y ver que en ese camino de estrés, trabajo y esfuerzo no me han regalado nada. Lo bueno del camino hecho, es que no lo he recorrido solo, y este es el momento de agradecérselo a todos.

En primer lugar agradecer a mi familia y amigos todo su apoyo e interés por la evolución de mi carrera. Especialmente a mi tía Lidia y mi prima Angélica. Gracias a mi padre, a mi madre y a mi hermana, por confiar en mí, apoyar mis decisiones, y darme siempre ánimos, cuando algo me preocupaba.

Gracias Pá por animarme cuando veía las cosas más difíciles, gracias Arjo por aguantarme, y gracias Má por preocuparte por mí, y en tantas noches que me quedaba trabajando hasta tarde, me preparases algo para comer y me lo acercases a la habitación. Os quiero.

Gracias Marta por estar a mi lado, confiar en mí, y apoyarme. Se que la espera ha sido dura, pero ves, al final todo llega, y después de todo, al final no ha sido tan larga ¿Verdad?. Espero que todo se compense y seguro que habrá merecido la pena. Te quiero.

Este camino está plagado de buenos momentos. Momentos compartidos con mis compañeros de clase, de los que se que saco dos amigos de verdad: Miguel y Carlos. Carlos, Miguel, ¡Sois muy grandes!

Tengo que agradecer también, grandes momentos a mis compañeros de beca_lega: Kike, Arti, Agui, Álvaro, Adri, Alba … ¡Qué lástima no haber solicitado la beca el primer año de universidad, y no el último! Sois gente auténtica.

Por último me gustaría dar las gracias a Jorge, mi tutor, por todas las ideas y valoraciones aportadas. Gracias por guiarme, y por tu compromiso conmigo y hacia este proyecto que considero un éxito. Hace cuatro meses no nos conocíamos, pero desde el primer día me transmitiste confianza, creo que has sido un tutor excelente, y eso se refleja en mi satisfacción por el proyecto.

Si de alguien me olvido, espero que me disculpe.
¡¡Muchas gracias a todos!!

Hoy termina una gran etapa de mi vida y comienza otra que espero sea al menos igual de exitosa.

Índice del documento:

Índice de figuras del documento:

Índice de tablas del documento

Capítulo 1. Introducción

1.1. Motivación del proyecto

El nacimiento de la telefonía móvil llega de la necesidad de estar comunicados con independencia del momento y el lugar en que nos encontremos.

Para cubrir esta necesidad, que fundamentalmente tenían los directivos de las grandes corporaciones, Motorola presenta en los años 80, el Motorola DynaTAC 8000x, el primer terminal de telefonía móvil, que marcaría un hito de gran relevancia en la historia de las telecomunicaciones (Figura 1).

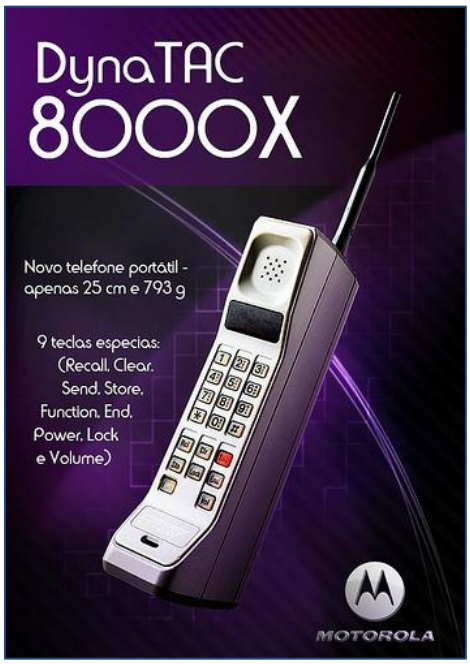

Figura 1. Anuncio promocional del DynaTAC 8000x.[1]

En la década de los 90, se marca un nuevo hito en la evolución de la telefonía móvil, con la llegada del GSM, y la digitalización de las comunicaciones, que mejora la calidad del servicio, y populariza el uso de la tecnología gracias al abaratamiento de los terminales.

En 2001 aparece la segunda generación y media de telefonía móvil, que fomenta el uso de Internet en los teléfonos móviles, paso previo a las redes de tercera generación, que trajeron la banda ancha a los dispositivos móviles, suponiendo otro gran hito en la historia de la telefonía móvil.

La llegada de la banda ancha, abrió un mundo de posibilidades, los terminales pasaron de ser simplemente teléfonos, con un pobre servicio de Internet, a convertirse en teléfonos inteligentes en los que la funcionalidad clásica de "teléfono" ha pasado a ser una funcionalidad más del terminal.

[1] Imagen obtenida en (DynaTAC 8000x)

En la Figura 2 puede verse le evolución tecnológica de los teléfonos móviles.

Figura 2. Evolución de los dispositivos móviles.[2]

En la actualidad se disfruta de la cuarta generación de telefonía móvil, que proporciona un valor altísimo de ancho de banda de entorno a los 100Mbits en movimiento, y 1Gbits en reposo (Khurshid, 2013), propiciando la aparición de nuevos dispositivos que permanecen conectados a la nube permanentemente.

Debido a la popularización de los Smartphones, así como de la variedad de aplicaciones que estos nos aportan, se almacena en ellos gran cantidad de datos: fotografías, números de teléfono, posiciones geográficas, contraseñas de banco, correos electrónicos, y un largo etcétera, haciendo que estos dispositivos puedan adquirir mucho más valor por su contenido que por su propio precio de venta.

Disponer de esta información tan valiosa, convierten a los teléfonos inteligentes en apreciados objetos de deseo, tanto para grandes multinacionales, como gobiernos o delincuentes informáticos.

La mayoría de aplicaciones tanto de escritorio, como las disponibles para teléfonos móviles, necesitan en algún momento de su ciclo de vida, almacenar cierta información que será utilizada en el futuro.

Las formas comunes en las que se puede encontrar información almacenada en los dispositivos móviles son: XML/plist, archivos binarios, y bases de datos SQLite, siendo estas últimas las más habituales para almacenar el grueso de la información.

Las bases de datos SQLite funcionan de forma similar a como lo hacen las bases de datos SQL tradicionales, por lo que comparten ciertas vulnerables, como por ejemplo los ataques de SQL Injection (SQLite Injection, 2014). La protección de las bases de datos ante este tipo de vulnerabilidades es fundamental, ya que dichos sistemas de almacenamiento pueden guardar información tan sensible como:

[2] Imagen obtenida en (Mobile Phone)

números de tarjetas o cuentas bancarias, mensajes confidenciales con planes de negocio, u otros datos tan privados como el lugar de residencia, o las posiciones geográficas de los últimos días. Como se puede concluir, la fuga de esta información puede suponer al propietario, un riesgo altísimo para sus intereses y su seguridad.

1.2. Objetivos

En este apartado se enumera una lista de objetivos marcados en este proyecto, acompañada de una descripción detallada de cada uno de ellos.

- Se desarrollará una aplicación móvil deliberadamente vulnerable a los ataques de SQL Injection, que nos permitirá adquirir conocimientos para realizar ataques de SQL Injection en bases de datos SQLite, y aprender a prevenirlos.
    - La aplicación deberá tener un ejercicio de hacking sobre un formulario de Login, en el que el usuario deberá acceder a un área restringida, con el usuario indicado, sin conocer los datos de acceso de este último.
    - La aplicación deberá tener un ejercicio de hacking sobre un formulario de búsqueda de productos, desde el cual deberá obtener la contraseña del usuario indicado.
    - La aplicación deberá tener un ejercicio de hacking sobre un formulario de inserción de productos, en el que el usuario de la aplicación consiga dar de alta más de un producto con una sola cumplimentación del formulario.
    - La aplicación deberá simular ser un programa de chat, en el que se implementará un ejercicio de hacking por el cual, el usuario deberá obtener el "token de usuario" del programa de chat.
    - La aplicación deberá simular ser un programa de chat, en el que se implementará un ejercicio de hacking por el cual, el usuario construirá una falsa conversación con su interlocutor, mediante mensajes enviados únicamente desde uno de los extremos.
    - La aplicación deberá ir mostrando ayudas que guíen al usuario hasta la consecución del ejercicio.
    - La finalidad de estos ejercicios es que los desarrolladores aprendan a proteger sus aplicaciones ante estos, o similares, tipos de ataques.

- Se realizará un script que permita analizar aplicaciones Android en búsqueda de una mala programación que permita la inyección de código SQL, de manera que este sirva para evaluar la seguridad de las aplicaciones Android, ante dicho tipo de ciberataques.
    - El script permitirá decompilar aplicaciones de la plataforma Android.
    - El script analizará los archivos decompilados, buscando vulnerabilidades frente a los ataques de SQL Injection.
    - El script generará un informe que con los resultados de los procesos anteriormente descritos, para su análisis posterior.

- Se realizará un estudio de varias aplicaciones de la plataforma oficial de Google, así como de otro tanto de aplicaciones malware y ver si se en estas se producen los fallos de programación comentados. Este estudio permitirá conocer si tanto desarrolladores como delincuentes informáticos emplean unas buenas prácticas de programación a la hora de desarrollar sus aplicaciones.

1.3. Contenido de la memoria

El documento presente se encuentra estructurado de la siguiente manera:

- **Capítulo 1: Introducción.** Se introduce al lector en el proyecto, presentándole la motivación y los objetivos propuestos de éste, así como la estructura de este documento.
- **Capítulo 2: Análisis.** Se presentará la situación actual en lo referente a lo estudiado en este proyecto, las alternativas de diseño valoradas, los casos de uso de las aplicaciones, junto con sus requisitos, restricciones y pruebas.
- **Capítulo 3: Diseño.** Se trata en detalle cómo se ha desarrollado el diseño de las aplicaciones, describiendo de manera detallada los componentes que las forman, incluyendo un diagrama de flujo.
- **Capítulo 4: Implementación.** Se mostrará el código fuente de la funciones principales de las aplicaciones del proyecto, así como el resultado de distintas pruebas presentadas en el capítulo de análisis.
- **Capítulo 5: Gestión del proyecto.** Se presentará la planificación del trabajo realizada al inicio del proyecto, contrastándola con el tiempo de esfuerzo finalmente realizado. Se enunciarán los medios técnicos empleados, y un breve análisis económico del proyecto.
- **Capítulo 6: Estudio.** Se detallarán los elementos del estudio y se analizarán sus resultados.
- **Capítulo 7: Conclusiones y líneas futuras.** Se presentara una conclusión final del proyecto, y la líneas de trabajo futuras relacionadas con dicho proyecto.
- **ANEXO 1: Manual de Aplicaciones.** Se presentará, de manera descriptiva, un manual de usuario, para la correcta explotación de ambas aplicaciones desarrolladas.
- **ANEXO 2: Listado de Aplicaciones Analizadas en el estudio.** Se enumerarán las aplicaciones utilizadas en el estudio.
- **ANEXO 3: Tabla de Resultados del Estudio.** Se presentará, en forma de tabla, los resultados obtenidos en el análisis de las aplicaciones.

Capítulo 2. Análisis

2.1. Introducción

En este segundo capítulo de la memoria del proyecto se comenzará tratando el estado de la cuestión, donde se analizarán las distintas alternativas de almacenamiento para dispositivos móviles, y las distintas aplicaciones similares disponibles en el mercado actual.

Seguidamente al estado de la cuestión, se analizarán las distintas alternativas de diseño planteadas para el desarrollo tanto de la aplicación móvil, como para la aplicación de escritorio, presentando una decisión justificada de porqué se elige cada una.

Ampliando este apartado, se encontraran documentados, los casos de uso, los requisitos, el marco regulador relacionado, las restricciones y las pruebas realizadas a las aplicaciones del proyecto.

2.2. Estado de la cuestión

En esta sección se presentará la problemática de almacenamiento en los dispositivos móviles, así como la situación actual de los proyectos que tratan dicho tema.

2.2.1. Alternativas de almacenamiento

La temática del proyecto trata la seguridad en el almacenamiento de la información en nuestros dispositivos móviles, cuya manera de almacenar la información, en cuanto a su formato se refiere, puede dividirse en cuatro tipos.

- Fichero de datos binario.
- Fichero XML.
- Base de datos SQLite.
- Almacenamiento remoto.

El almacenamiento remoto se diferencia claramente del resto, debido a su naturaleza, con la que la información simularía estar almacenada en el dispositivo, pero que en realidad se encontraría disponible en algún servidor accesible por el dispositivo móvil.

Por archivos binarios, y teniendo en cuenta (Binary file, 2014), se podría entender que fueran todos aquellos que no correspondieran con los otros tres enumerados arriba. Estos ficheros serían por tanto los archivos de imagen, audio, texto, etc.

Para almacenar las preferencias de las aplicaciones es muy habitual que se utilicen ficheros de tipo XML, debido a su gran versatilidad al tratarse de un lenguaje de etiquetado.

Por último, las bases de datos SQLite, suponen el sistema de almacenamiento principal de los dispositivos móviles, debido entre otros motivos, a su uso eficiente

de la memoria, su naturaleza auto-contenida, o su falta de necesidad de mantenimiento por parte de un administrador (SQLite features, 2014).

Respecto al tema de la seguridad de estos archivos, todos comparten vulnerabilidades de confidencialidad e integridad si el dispositivo ha sido sustraído. Sin embargo se puede intentar proteger la seguridad de estos datos con medidas tan sencillas como las de cifrar el contenido del dispositivo, y bloquear el acceso a los datos mediante el uso de contraseñas.

Como vulnerabilidad particular que sufren los archivo SQLite debido a su diseño, es la de que mediante el uso de consultas maliciosas, es posible acceder a cierta información que no debería dejar conocer. Esta vulnerabilidad es conocida cómo SQL Injection (SQLite Injection, 2014), y es la que se va a tratar de profundizar en este proyecto.

En la Figura 3 se puede ver representado el modelo de datos presentado más arriba, tal como se plantea en la plataforma Android.

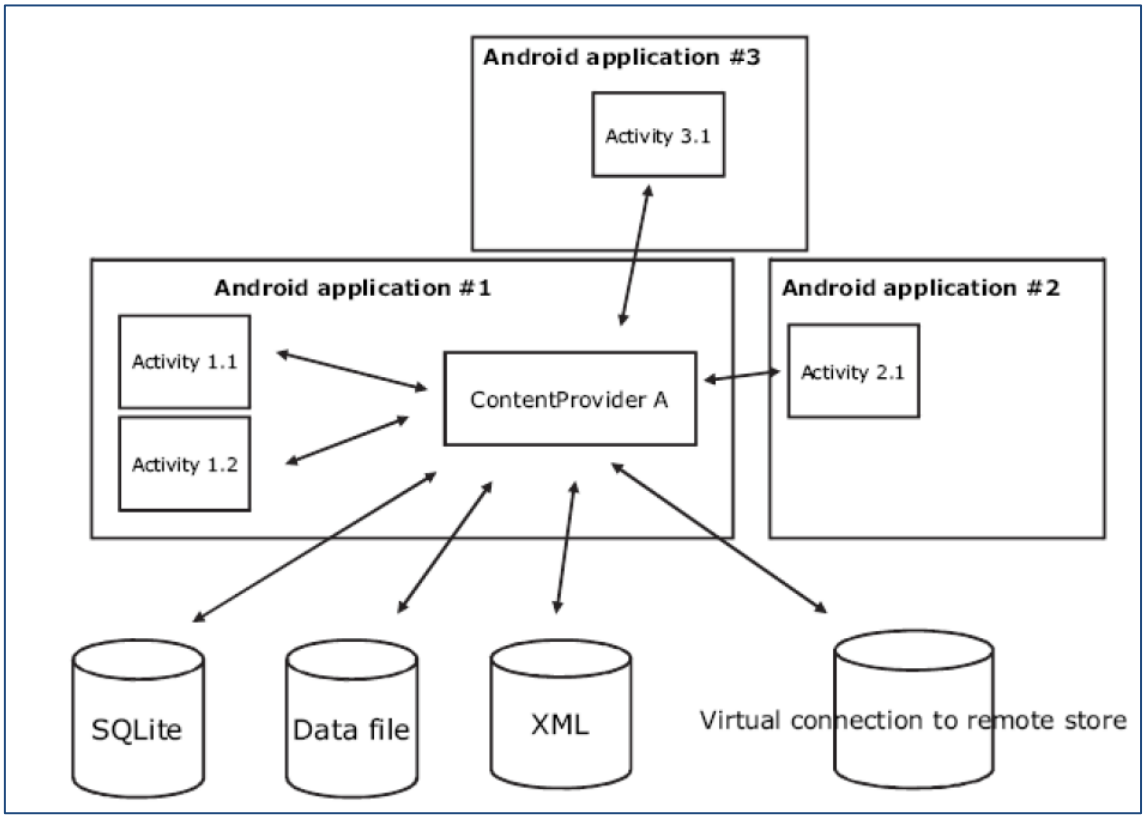

Figura 3. Modelo de Datos en Android[3]

Dado que SQLite se establece como formato principal para el almacenamiento de información tal como acabamos de ver, ha decidido centrarse el estudio en la seguridad de este.

[3] Imagen obtenida en (Modelo de Datos Android)

2.2.2. Herramientas similares de concienciación

Las aplicaciones para concienciar a los programadores, de implementar aplicaciones seguras, son muy típicas en el mundo del desarrollo de programas informáticos, y existen aplicaciones tales como WebGoat (OWASP WebGoat, 2013), iGoat (OWASP iGoat, 2014), o DroidGoat (OWASP DroidGoat, 2014), que tratan de llevar a cabo este principio, y que se prevenga a nuestras aplicaciones de multitud ataques maliciosos, de las que éstas, con gran seguridad, serán objeto.

En cuanto a programas que busquen las vulnerabilidades de SQL lnjection, existen en el mercado software como WebCruiser (WebCruiser, 2014) o Netsparker (Netsparker, 2014), que analizan aplicaciones web, notificando de las vulnerabilidades encontradas.

A continuación se enumeran las aplicaciones e iniciativas recién mencionadas, entrando más en detalle en cada una de ellas.

- **WebGoat**

  WebGoat es una aplicación web, de código abierto y gratuita, administrada por OWASP (OWASP, 2014), que ha sido diseñada deliberadamente con graves vulnerabilidades de seguridad, y cuya finalidad es servir como plataforma para impartir lecciones de ciberseguridad.

  En cada una de las actividades, los usuarios deberán demostrar su conocimiento sobre el tema, mediante la explotación de la vulnerabilidad correspondiente. Un ejemplo de actividad en WebGoat es el representado en la Figura 4.

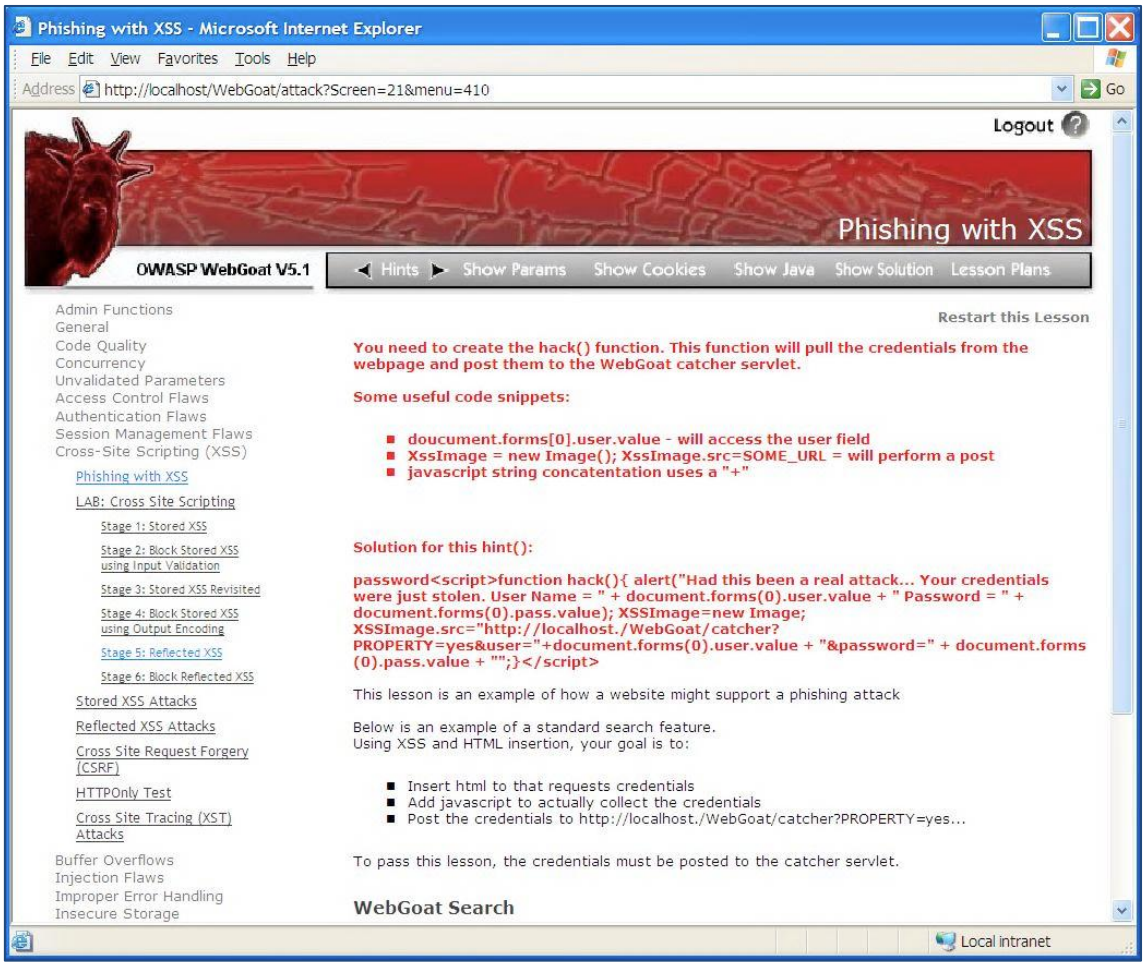

Figura 4. Ejemplo de Actividad en WebGoat[4].

Según los autores, la seguridad en aplicaciones web es difícil tanto de aprender como de practicar, ya que buscar vulnerabilidades, sin permiso del propietario de la web, no es legal, ni éticamente correcto, y ni siquiera se asegura de verificar conocimientos, ya que no se sabe de antemano que vulnerabilidades contiene la aplicación.

El objetivo buscado por los autores, es que el proyecto WebGoat se convierta en un entorno educativo de-facto para la seguridad de aplicaciones web, proporcionando un entorno cerrado y controlado en el que se puedan explotar legalmente sus vulnerabilidades, las cuales son bien conocidas previamente.

El proyecto WebGoat cuenta con más de 30 lecciones de temáticas distintas, entre las que destacan:

- Cross-Site Scripting.
- Control de accesos.
- Seguridad en hilos de procesos.
- Manipulación de campos ocultos en formularios.
- Manipulación de parámetros.
- Cookies de sesión débiles.

[4] Imagen obtenida en (OWASP WebGoat, 2013)

- Inyección SQL a ciegas.
- Inyección SQL numérica.
- Inyección SQL en cadenas de texto.
- Servicios Web.
- Fallos en proceso de autenticación.
- Peligros en el uso de comentarios HTML.

De cara al futuro, otro objetivo que persiguen desde OWASP, es que el proyecto WebGoat consiga derivarse en un sistema Honeypot (Yeh, 2008) que registre todos los ataques recibidos y que sirva a los encargados de la ciberseguridad de las empresas, para mejorar la seguridad de su red de trabajo.

- **iGoat**

Con WebGoat como referente, iGoat se establece como herramienta de aprendizaje para desarrolladores de la plataforma iOS, de código abierto y gratuita.

Al igual que WebGoat, esta aplicación está organizada en actividades, donde los principales módulos del temario son los enumerados en el Top 10 de Riesgos en dispositivos móviles según OWASP (OWASP Top 10 Mobile Risks, 2014). Dichos módulos serían los ilustrados en la Figura 5:

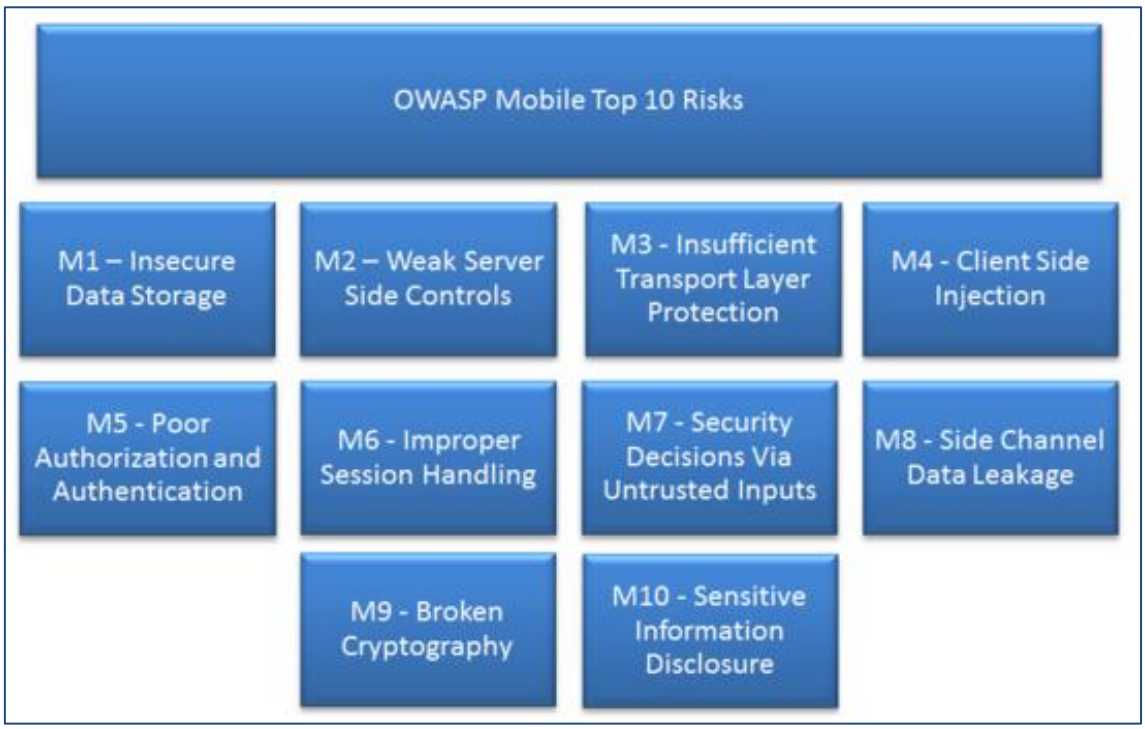

Figura 5. Top 10 de Riesgos en Móviles según OWASP[5].

El flujo de trabajo de cada una de las lecciones de iGoat es el siguiente:

1. Introducción a la vulnerabilidad.
2. Explotación de la vulnerabilidad.

[5] Imagen obtenida en (OWASP Top 10 Mobile Risks, 2014)

3. Descripción de posibles soluciones.
4. Corrección del problema y recompilado de la aplicación.

En la Figura 6 se puede observar un ejemplo una de las actividades comprendidas en iGoat.

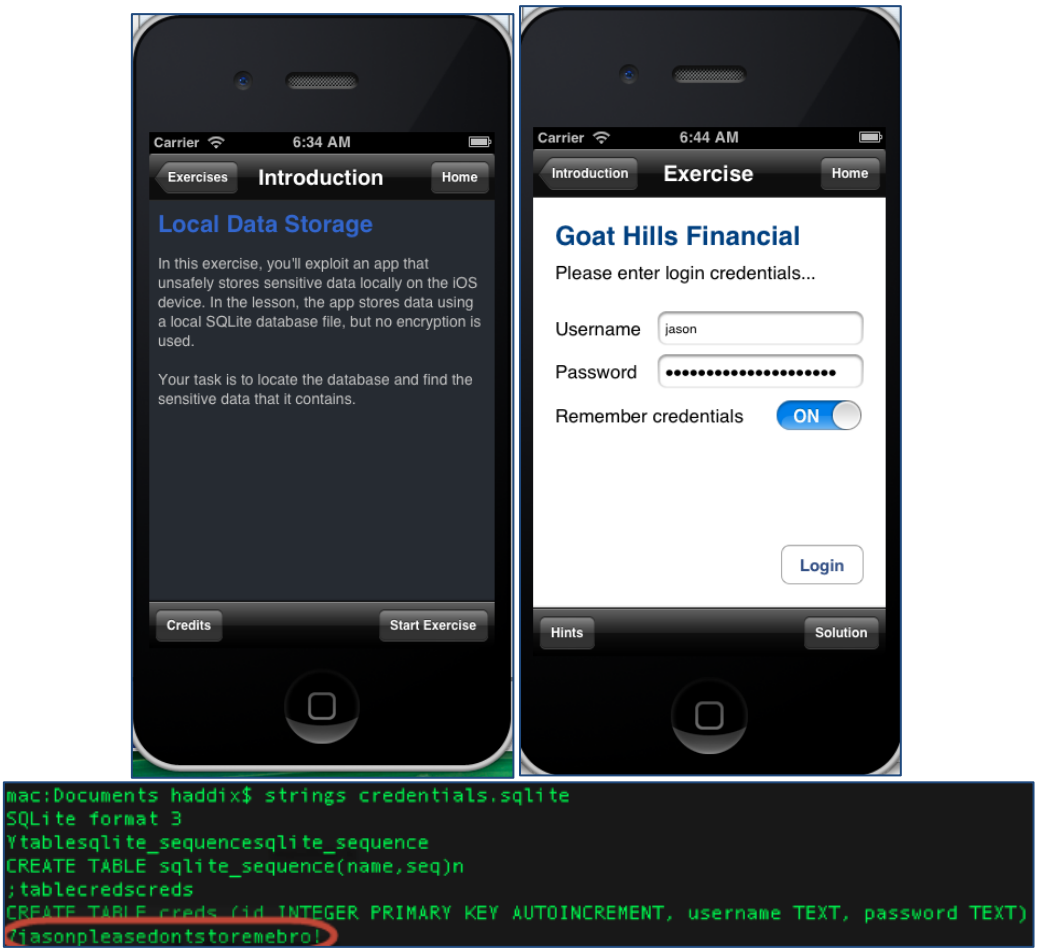

Figura 6. Ejemplo de actividad en iGoat[6].

En definitiva, el objetivo que se propone con iGoat, no es otro que los desarrolladores de aplicaciones para los dispositivos móviles de Apple, adquieran los conocimientos necesarios para realizar una correcta programación de sus aplicaciones, haciendo que estas sean menos vulnerables a los ataques de delincuentes informáticos.

- **DroidGoat**

La propuesta de DroidGoat es la misma que la de iGoat, en esta ocasión orientada a la programación de dispositivos Android, pero en contraposición a iGoat, en el momento de la elección de este proyecto, la línea de desarrollo de DroidGoat se encuentra desatendida.

[6] Imágenes obtenidas en (Ejemplo iGoat)

- **WebCruiser**

WebCruiser es una herramienta para desarrolladores y pentesters (Pentesting, 2013), que analiza aplicaciones web en busca de vulnerabilidades, ante ataques tales como: SQL Injection, Cross Site Scripting, o XPah Injection entre otros. Una vez realizados todos los procesos de análisis, la aplicación ofrece un informe pormenorizado de los resultados obtenidos, tal como se refleja en la Figura 7.

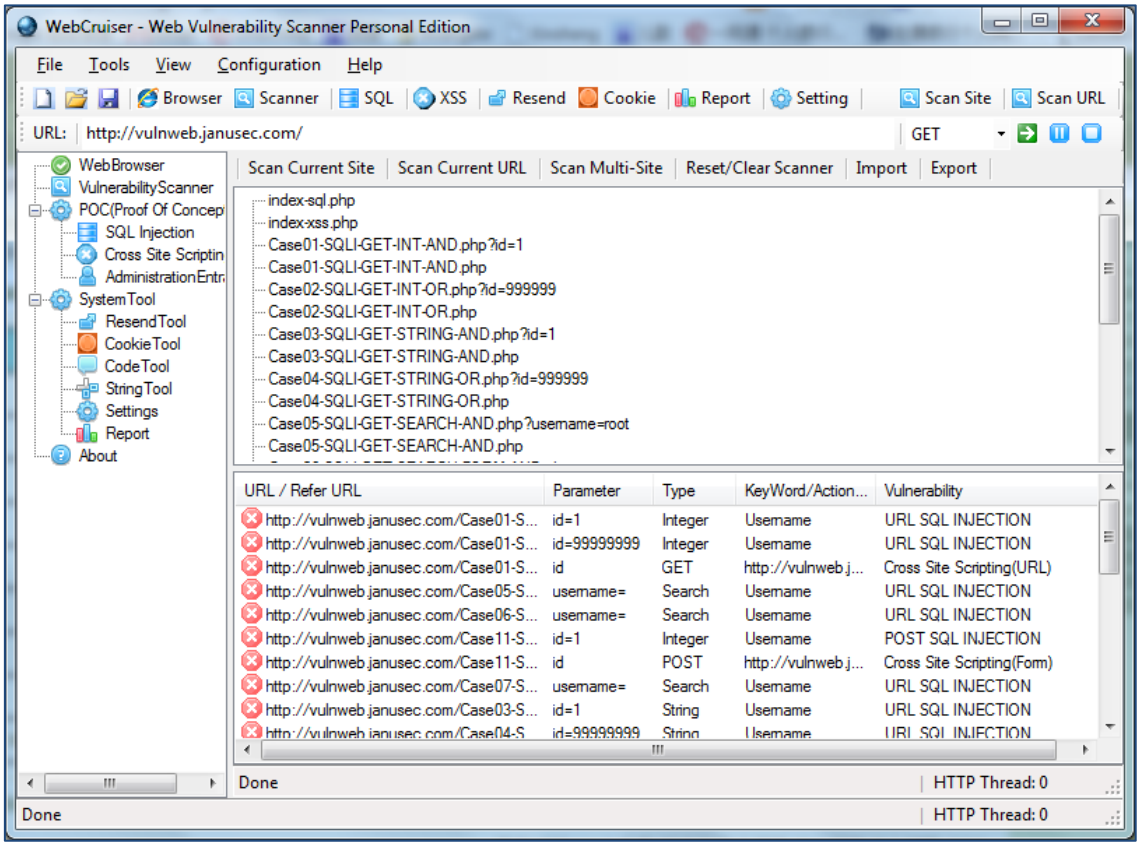

Figura 7. Captura de la Herramienta WebCruiser[7].

La herramienta WebCruiser es multiplataforma, y se puede encontrar disponible para su compra, en su página web, con formato para Windows, Mac OS X, e iOS.

- **Netsparker**

Netsparker es al igual que WebCruiser un analizador de aplicaciones web de pago, orientado a la búsqueda de sus debilidades. Según sus creadores, es capaz de encontrar una gran cantidad de vulnerabilidades distintas entre las que destacan SQL Injection, Cross-Site Scripting, Command Injection, o inserción de ficheros tanto locales como remotos.

Terminado el proceso de análisis, la herramienta genera un informe detallado de las vulnerabilidades encontradas, como se muestra en la Figura 8, para que estas puedan ser corregidas tal como sugieren con sus comentarios aportados.

[7] Imagen obtenida en (WebCruiser)

A diferencia de WebCruiser, Nerstarker no es una aplicación multiplataforma, ya que la única plataforma soportada para su instalación, es el sistema operativo de Microsoft.

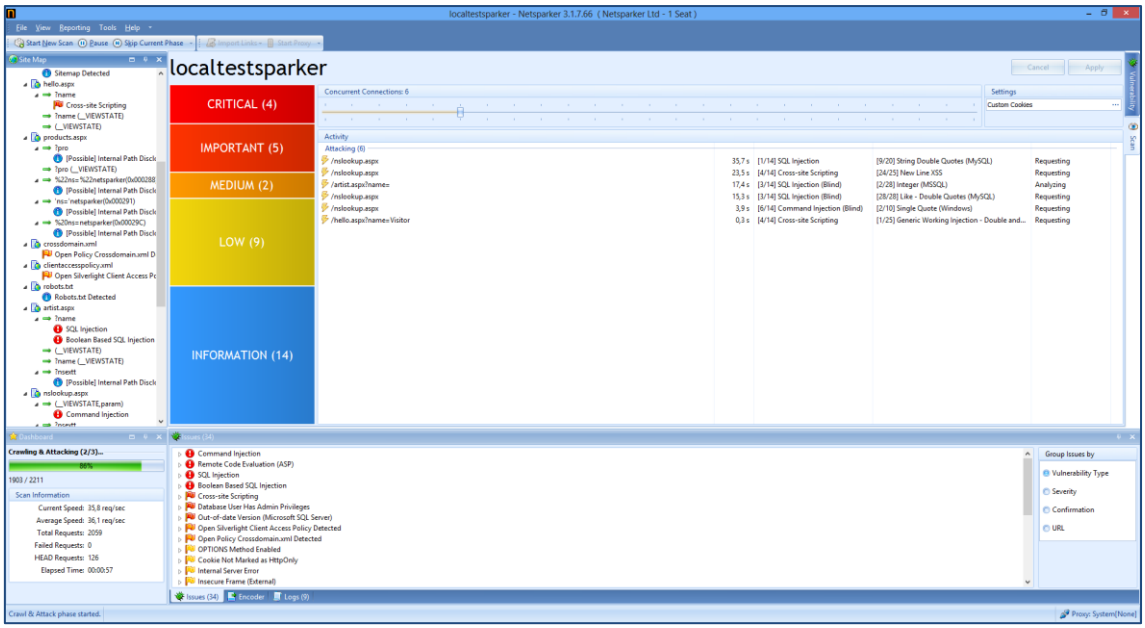

Figura 8. Captura de Netsparker[8].

2.3. Alternativas de diseño

En este apartado se estudiarán las distintas posibilidades en las que desarrollar las aplicaciones del proyecto, concluyendo con una decisión justificada de las plataformas elegidas.

2.3.1. Valoración para plataforma Smartphone

Entre las plataformas de teléfonos inteligentes, con mayor cuota de mercado en el momento actual, tal como confirman los informes de Garthner en la Figura 9, encontramos los sistemas operativos: Windows Phone, iOS, Android o BlackBerry.

[8] Imagen obtenida en (Netsparker)

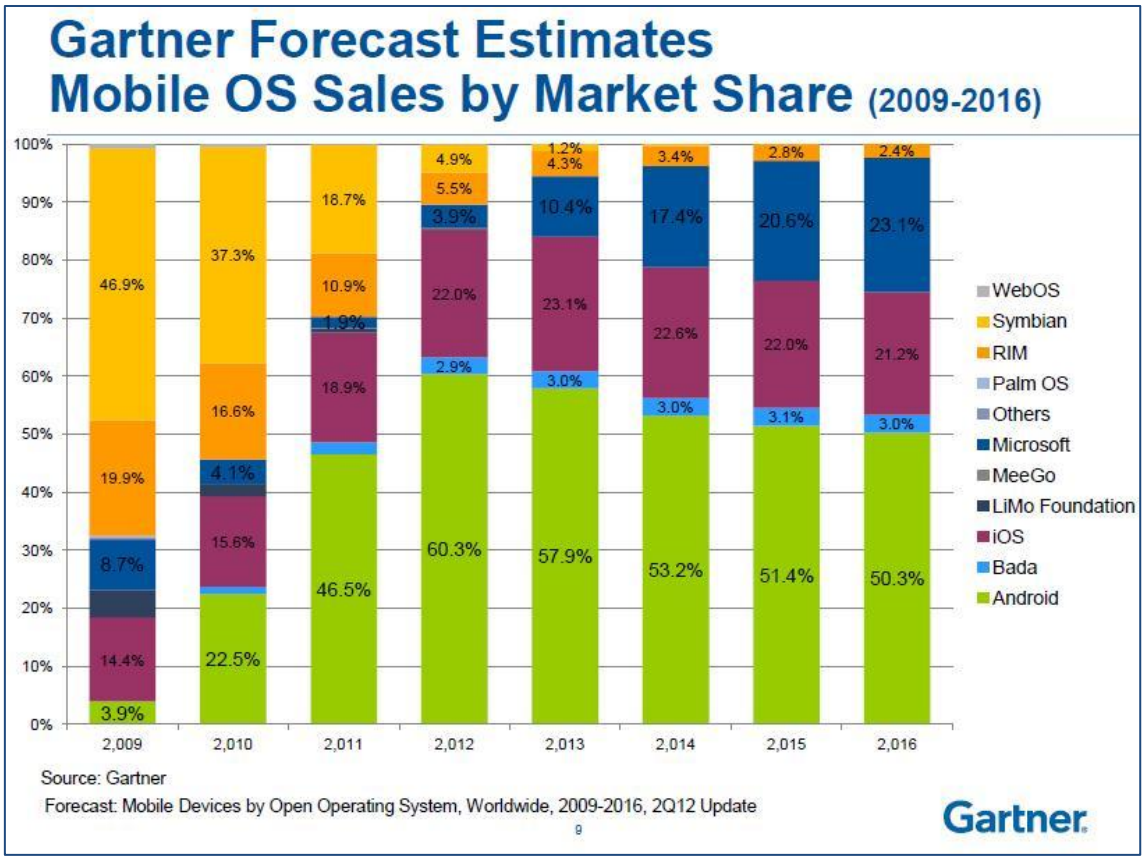

Figura 9. Previsión de Mercado en Sistemas Operativos Móviles Según Garther[9].

Para valorar las distintas plataformas, y tomar una decisión justificada sobre que sistema operativo para el que desarrollar la aplicación, se compararán los siguientes puntos:

- Penetración en el mercado.
- Rapidez de aprendizaje del lenguaje.
- Facilidad de programación.
- Existencia de aplicaciones similares en la misma plataforma.

Todos los puntos se valorarán de 1 a 10 en función de la experiencia personal, sin embargo su ponderación será distinta. La penetración en el mercado tendrá una valoración del 30%, la rapidez en el aprendizaje del lenguaje tendrá un peso del 15%, la facilidad de programación un 20%, y por último la existencia previa de aplicaciones similares en la misma plataforma un 35% de importancia.

En cuanto a lo referente a la puntuación, la penetración en el mercado tendrá un valor mayor, cuantos más dispositivos existan en el mercado, la rapidez de aprendizaje tendrá una puntuación más alta cuando su aprendizaje sea más sencillo, la puntuación para la facilidad de programación tomará un valor más alto cuanto más sencillo de desarrollar sea, y para terminar, la existencia de aplicaciones similares puntuará más, cuando no existan aplicaciones en el mercado.

[9] Imagen obtenida en (Forbes, 2013)

Penetración en el mercado.
En función de la cantidad de dispositivos vendidos según su sistema operativo se han otorgado los siguiente valores.

	Windows Phone 8	iOS	Android	BlackBerry
Penetración en el mercado	2	5	8	1
Total Ponderado (30%)	0,6	1,5	2,4	0,3

Tabla 1. Penetración en el Mercado.

Rapidez de aprendizaje del lenguaje.
Valorando el esfuerzo que se debe hacer para aprender los lenguajes de programación con los que habría que desarrollar la aplicación se han asignado los siguientes valores. (Windows Phones: C#, iOS: Objetive-C, Andoid: Java, BlackBerry: Java)

	Windows Phone 8	iOS	Android	BlackBerry
Rapidez de aprendizaje del lenguaje	6	6	9	9
Total Ponderado (15%)	0,9	0,9	1,35	1,35

Tabla 2. Rapidez de Aprendizaje del Lenguaje.

Facilidad de programación.
Teniendo en cuenta la plataforma de desarrollo, y la complejidad para implementar soluciones con cada una de las plataformas, han correspondido las siguientes puntuaciones.

	Windows Phone 8	iOS	Android	BlackBerry
Facilidad de programación	5	5	7	6
Total Ponderado (20%)	1	1	1,4	1,2

Tabla 3. Facilidad de Programación.

Existencia de aplicaciones similares en la misma plataforma.
Valorando la existencia o no, y el grado de desarrollo de aplicaciones similares en el mismo sistema operativo, se han asignado los valores que aparecen en esta tabla.

	Windows Phone 8	iOS	Android	BlackBerry
Existencia de aplicaciones similares en la misma plataforma	10	1	6	10
Total Ponderado (35%)	3,5	0,35	2,1	3,5

Tabla 4. Existencia de aplicaciones similares en la misma plataforma.

Resumen general de resultados:

Totales ponderados	Windows Phone 8	iOS	Android	BlackBerry
Penetración en el mercado (30%)	0,6	1,5	2,4	0,3
Rapidez de aprendizaje del lenguaje (15%)	0,9	0,9	1,35	1,35
Facilidad de programación (20%)	1	1	1,4	1,2
Existencia de aplicaciones similares en la misma plataforma (35%)	3,5	0,35	2,1	3,5
TOTAL	**6**	**3,75**	**7,25**	**6,35**

Tabla 5. Resumen General de Resultados para la Aplicación en Smartphone.

Tal como muestra la suma total de los resultados, se determina que la plataforma más adecuada para el desarrollo de la aplicación es Android. Destaca claramente a raíz de la gran porción del mercado de dispositivos móviles que controla, el poco esfuerzo de aprendizaje necesario, y la baja dificultad de programación desde el punto de vista del autor de este proyecto.

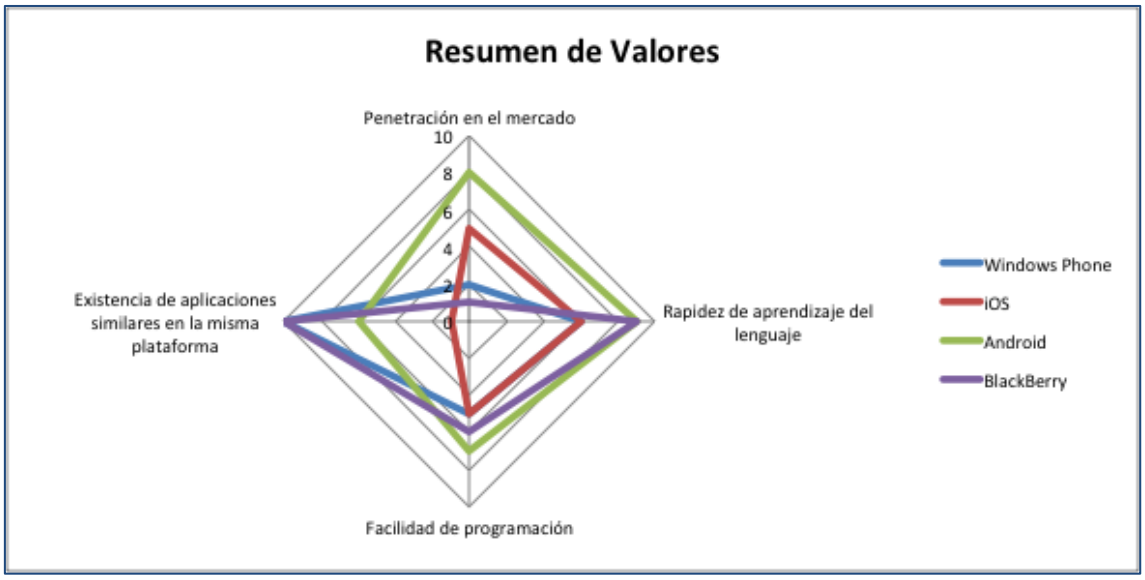

Figura 10. Gráfico con Resumen de Valores para la Aplicación en Smartphone.

En la Figura 10 se puede ratificar los resultados obtenidos, observando rápidamente y sin necesidad de hacer números, que el sistema operativo con valores más altos y estables es Android.

2.3.2. Valoración para plataforma PC

El desarrollo de la aplicación de análisis de vulnerabilidades, se plantea posterior a la decisión de implementar la aplicación de aprendizaje sobre SQL Injection en dispositivos Android.

Para la implementación de este programa, se tomará como base la herramienta de compilación de aplicaciones Android: Apktool (Apktool, 2014). Esta aplicación se ejecuta desde línea de comandos, y es compatible con Windows, Linux, y Mac OS X. Es una herramienta gratuita, y su código fuente puede ser descargado desde su página oficial.

La funcionalidad de Apktool, y por la cual es la piedra angular de esta segunda aplicación, es la de decompilar y volver a compilar programas del sistema operativo Android.

Establecida la premisa de emplear la herramienta Apktool dentro de la segunda aplicación, se debe valorar que leguaje de programación es más adecuado para su implementación.

Las opciones valoradas en esta ocasión han sido tres bien conocidas por el desarrollador, lenguaje C, Java, o Shell Script de Linux.

Tal como se hizo con el caso anterior, para llegar a un resultado justificado, se van a puntuar una serie de condiciones, que evalúen que opción es la más adecuada.

Los puntos que se van a valorar son:

- Facilidad de integración con Apktool.

- Facilidad de programación de funcionalidades de búsqueda.
- Facilidad para generar un informe.

Todos los puntos se valorarán de 1 a 10 en función de la experiencia personal, sin embargo su ponderación será distinta. La facilidad de integración con Apktool tendrá una valoración del 40%, la facilidad de programación de funcionalidades de búsqueda tendrá un peso del 30%, y por último la facilidad para generar un informe representará un 30% de importancia.

En cuanto a lo referente a la puntuación, la facilidad de integración con Apktool tendrá un valor mayor, cuanto más fácil sea su unión, la facilidad de programación de funcionalidades de búsqueda tendrá una puntuación más alta cuando su implementación sea más sencilla, y para terminar, la facilidad para generar un informe puntuará más, cuando suponga menos esfuerzo su programación.

Facilidad de integración con Apktool.
En este punto se valora la complejidad a la hora de integrar Apktool con las correspondientes opciones.

	C Language	Java	Shell Script
Facilidad de integración con Apktool	7	7	10
Total Ponderado (40%)	2,8	2,8	3

Tabla 6. Facilidad de Integración con Apktool.

Facilidad de programación de funcionalidades de búsqueda.
En este punto se mide la dificultad para implementar funciones que busquen el código fuente de la aplicación fallos de programación que deriven en vulnerabilidades en la seguridad de la aplicación.

	C Language	Java	Shell Script
Facilidad de programación de funcionalidades de búsqueda	6,5	6,5	8
Total Ponderado (30%)	1,95	1,95	2,4

Tabla 7. Facilidad de Programación de Funcionalidades de Búsqueda.

La razón por la que Shell Script tenga una valoración tan alta es debido a la existencia del comando grep, que se encarga precisamente de buscar cadenas de texto dentro de ficheros.

Facilidad para generar un informe.
En este punto, el objetivo es medir el esfuerzo en la creación de un informe detallado con los resultados obtenidos en el proceso de búsqueda de vulnerabilidades.

	C Language	Java	Shell Script
Facilidad para generar un informe	7	7	9
Total Ponderado (30%)	2,1	2,1	2,7

Tabla 8. Facilidad para Generar un Informe.

Resumen general de resultados:

Totales ponderados	C Language	Java	Shell Script
Facilidad de integración con Apktool (40%)	2,8	2,8	3
Facilidad de programación de funcionalidades de búsqueda (30%)	1,95	1,95	2,4
Facilidad para generar un informe (30%)	2,1	2,1	2,7
TOTAL	**6,85**	**6,85**	**8,1**

Tabla 9. Resumen General de Resultados para la Herramienta de Búsqueda.

Si se analiza la suma total de los resultados, se llega a la clara conclusión de que la forma más sencilla y rápida de implementar el sistema de búsqueda de vulnerabilidades, es mediante la programación de un Shell Script de Unix. La elección del Script es contundente, ya que se puede observar que en todos los ítems de valoración, su puntuación es la más elevada.

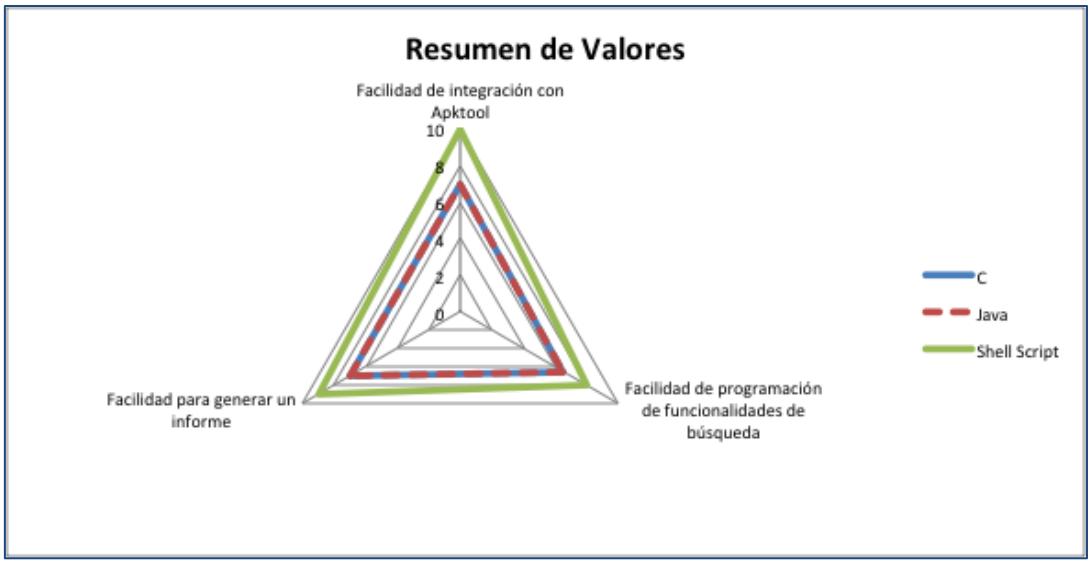

Figura 11. Gráfico con Resumen de Valores para la Herramienta de Búsqueda.

En la Figura 11 se puede ratificar los resultados obtenidos, observando rápidamente y sin necesidad de hacer números, que la forma más sencilla de implementar la aplicación de búsqueda de vulnerabilidades, es mediante el desarrollo de un Shell Script.

2.4. Casos de Uso

En el siguiente apartado se podrán conocer los distintos casos de uso disponibles tanto para la aplicación Android como para el Script de análisis.

2.4.1. Casos de uso para la aplicación Android

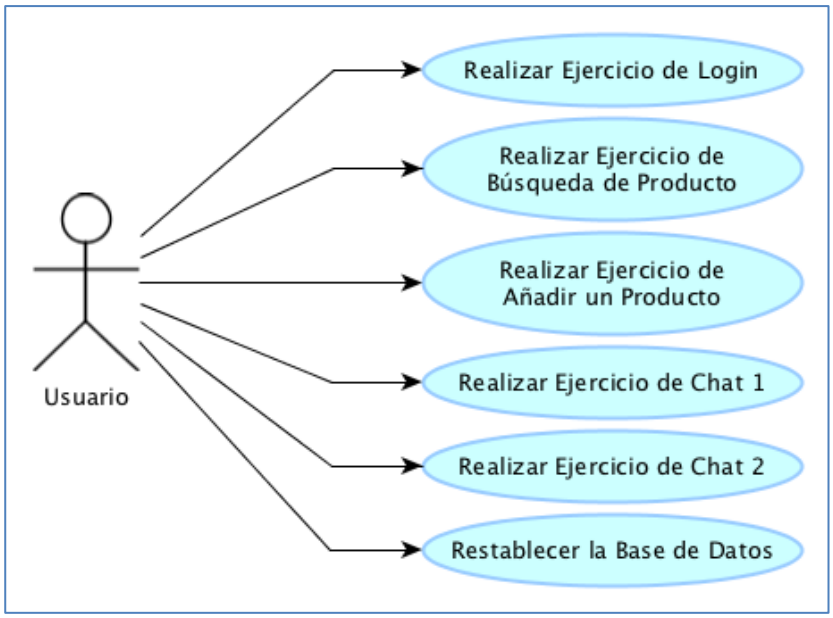

Figura 12. Diagrama de Casos de Uso en Aplicación Móvil.

Tal como se ve en la Figura 12, las acciones que puede realizar el usuario en la aplicación móvil son: realizar los ejercicios de login, búsqueda de producto, incorporación de un nuevo producto, dos ejercicios basados en una aplicación de chat, y por último, restablecer la base de datos.

Para detallar un poco más cada uno de los casos de uso, se presentan las siguientes tablas.

ID	
Nombre	
Objetivo	
Descripción	
Pre-condiciones	
Post-condiciones	
Escenario	
Causas de fallo	

Tabla 10. Tabla de Ejemplo para Casos de Uso.

- **ID:** Código identificativo único con el formato CU-X-YY. La X representa con "A", a la aplicación Android, y con "S", a la aplicación de Búsqueda. Las YY representarán un numero correlativo iniciado en 00.
- **Nombre:** Nombre descriptivo para el caso de uso.
- **Objetivo:** Presentación de la finalidad del caso de uso.
- **Descripción:** Explicación de los procesos establecidos en cada caso de uso.
- **Pre-condiciones:** Condiciones previas para realizar el caso de uso.
- **Post-condiciones:** Condiciones de salida para una realización correcta del caso de uso.
- **Escenario:** Enumeración de pasos que componen el caso de uso.
- **Causas de fallo:** Posibles situaciones problemáticas que impidan la correcta ejecución del caso de uso.

ID	CU-A-01
Nombre	Realizar el ejercicio de login.
Objetivo	Entrar en el desafío, y acceder como administrador, sin conocer el usuario ni la contraseña, en el área restringida de este.
Descripción	El usuario rellena el formulario con un usuario y contraseña elegido por él, accediendo al área restringida.
Pre-condiciones	Introducción de datos en todos los campos del formulario, y que los datos introducidos sean los correctos para poder entrar como usuario administrador.
Post-condiciones	La aplicación informa al usuario de que ha completado el desafío.
Escenario	1. El usuario acepta el desafío. 2. El usuario introduce una combinación correcta de datos

ID	CU-A-01
	en los campos de usuario y contraseña. 3. La aplicación informa al usuario de que ha conseguido acceder como usuario administrador mediante un hacking del sistema.
Causas de fallo	No rellenar todos los campos del formulario. Cumplimentar el formulario con datos incorrectos.

Tabla 11. CU-A-01, Realizar el Ejercicio de Login.

ID	CU-A-02
Nombre	Realizar el ejercicio de búsqueda de producto.
Objetivo	Entrar en el desafío, y obtener la contraseña de usuario administrador, perteneciente a otra tabla distinta a la de productos en la base de datos.
Descripción	El usuario rellena el campo de búsqueda con una serie de instrucciones que le permite obtener la contraseña del usuario administrador.
Pre-condiciones	Introducción de datos correctos para poder recuperar la contraseña del administrador.
Post-condiciones	La aplicación informa al usuario de que ha completado el desafío.
Escenario	1. El usuario acepta el desafío. 2. El usuario introduce una combinación correcta de datos en el campo de búsqueda. 3. La aplicación informa al usuario de que ha conseguido obtener la contraseña del administrador mediante un hacking del sistema.
Causas de fallo	Cumplimentar el formulario con datos incorrectos.

Tabla 12. CU-A-02, Realizar el Ejercicio de Búsqueda de Producto.

ID	CU-A-03
Nombre	Realizar el ejercicio de añadir un producto.
Objetivo	Entrar en el desafío, y añadir más de un producto en la en la base de datos, mediante una sola cumplimentación del formulario.
Descripción	El usuario rellena el formulario de incorporación de productos, con una serie de instrucciones que le permite añadir más de un producto con un solo envío del formulario.
Pre-condiciones	Introducción de datos en todos los campos del formulario, y que los datos introducidos sean los correctos para poder ingresar más de un producto a la vez.
Post-condiciones	La aplicación informa al usuario de que ha completado el desafío.
Escenario	1. El usuario acepta el desafío. 2. El usuario introduce una combinación correcta de datos en el formulario.

ID	CU-A-03
	3. La aplicación informa al usuario de que ha conseguido añadir varios productos a la base de datos, mediante un hacking del sistema.
Causas de fallo	No rellenar todos los campos del formulario. Cumplimentar el formulario con datos incorrectos.

Tabla 13. CU-A-03, Realizar el Ejercicio de Añadir un Producto.

ID	CU-A-04
Nombre	Realizar el ejercicio de chat 1.
Objetivo	Entrar en el desafío, y mandar el mensaje correcto para obtener el token de usuario.
Descripción	El usuario envía un mensaje a su interlocutor, con una serie de instrucciones que le permite obtener su token de usuario.
Pre-condiciones	Introducción de los datos correctos para poder obtener el token de usuario.
Post-condiciones	La aplicación informa al usuario de que ha completado el desafío.
Escenario	1. El usuario acepta el desafío. 2. El usuario envía un mensaje con el contenido correcto que le permite obtener el token de usuario. 3. La aplicación informa al usuario de que ha conseguido obtener el token, mediante un hacking del sistema.
Causas de fallo	Enviar un mensaje con datos incorrectos.

Tabla 14. CU-A-04, Realizar el Ejercicio Chat 1.

ID	CU-A-05
Nombre	Realizar el ejercicio de chat 2.
Objetivo	Entrar en el desafío, y mandar un mensaje que parezca que haya sido enviado por el interlocutor, y cuyo contenido sea “you have been hacked”.
Descripción	El usuario envía un mensaje a su interlocutor, con una serie de instrucciones que permite simular ser un mensaje enviado por el interlocutor, y cuyo contenido es “you have been hacked”.
Pre-condiciones	Introducción de los datos correctos para poder simular un mensaje enviado por el interlocutor, y que el contenido del mensaje sea “you have been hacked”.
Post-condiciones	La aplicación informa al usuario de que ha completado el desafío.
Escenario	1. El usuario acepta el desafío. 2. El usuario envía un mensaje con el contenido correcto que le permite simular un mensaje enviado por el interlocutor, y cuyo contenido es “you have been hacked”. 3. La aplicación informa al usuario de que ha conseguido superar el desafío, mediante un hacking del sistema.

ID	CU-A-05
Causas de fallo	Enviar un mensaje con datos incorrectos.

Tabla 15. CU-A-05, Realizar el Ejercicio Chat 2.

ID	CU-A-06
Nombre	Restablecer la base de datos.
Objetivo	Recuperar el contenido de la base de datos en su estado original.
Descripción	El usuario solicita a la aplicación que restablezca la base de datos, y ésta borra todo su contenido, cargando de nuevo la información original.
Pre-condiciones	Localizar el botón en el menú de opciones.
Post-condiciones	La aplicación informa al usuario de que ha restablecido la base de datos.
Escenario	1. El usuario envía la petición a la aplicación. 2. La aplicación borra el contenido de toda la base de datos y lo regenera de nuevo. 3. La aplicación informa al usuario de que se ha restablecido la base de datos.
Causas de fallo	Versión de la base de datos distinta de la aplicación utilizada.

Tabla 16. CU-A-06, Restablecer la Base de Datos.

2.4.2. Casos de uso para Shell Script

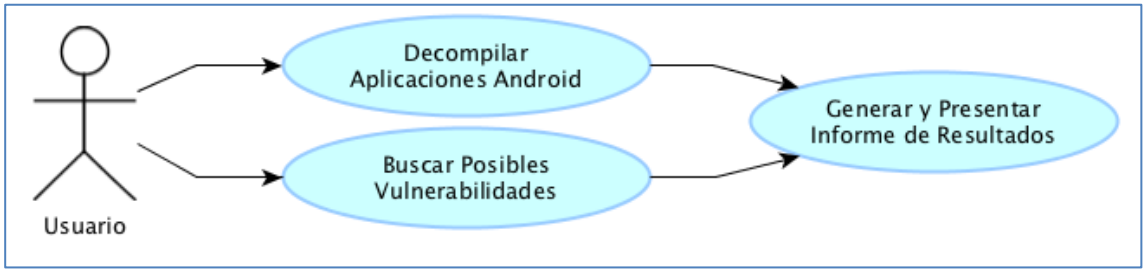

Figura 13. Diagrama de Casos de Uso en Shell Script

El diagrama de casos de usos para la aplicación de búsqueda de vulnerabilidades, representado con la Figura 13, presenta las dos funcionalidades principales y accesibles por el usuario, que son la decompilación de aplicaciones Android, y la búsqueda de vulnerabilidades SQL Injection dentro del código decompilado de dichas aplicaciones, por último se muestra la funcionalidad común a las dos principales, que es la generación y presentación de un informe de resultados.

ID	CU-S-01
Nombre	Decompilar aplicaciones Android.
Objetivo	Obtener el código fuente de las aplicaciones Android.
Descripción	El usuario introduce el comando –d acompañado del directorio con las aplicaciones Android, y el programa las decompila.

ID	CU-S-01
Pre-condiciones	Se debe elegir la opción –d, y el directorio de aplicaciones debe existir y contener aplicaciones.
Post-condiciones	La aplicación lanza la función de generar y presentar informes.
Escenario	1. El usuario introduce por línea de comando la opción –d y el directorio con las aplicaciones. 2. La aplicación decompila los archivos, y lanza la función de generar y presentar informes.
Causas de fallo	No introducir los parámetros correctos. El directorio de aplicaciones está vacío. Los archivos del directorio de aplicaciones no son ficheros apk[10] o compatibles. No tener permisos de escritura sobre el directorio de ejecución.

Tabla 17. CU-S-01, Descompilar Aplicaciones Android.

ID	CU-S-02
Nombre	Buscar vulnerabilidades SQL Injection.
Objetivo	Buscar posibles vulnerabilidades SQL Injection en el código de las aplicaciones.
Descripción	El usuario introduce el comando –s acompañado del directorio con las aplicaciones descompiladas de Android, y el programa busca las posibles vulnerabilidades.
Pre-condiciones	Se debe elegir la opción –s, y el directorio de aplicaciones decompiladas debe existir y contener aplicaciones.
Post-condiciones	La aplicación lanza la función de generar y presentar informes.
Escenario	1. El usuario introduce por línea de comando la opción –s y el directorio con las aplicaciones decompiladas. 2. La aplicación busca las posibles vulnerabilidades, y lanza la función de generar y presentar informes.
Causas de fallo	No introducir los parámetros correctos. El directorio de aplicaciones decompiladas está vacío. Los archivos del directorio de aplicaciones no tienen permisos de lectura para la aplicación.

Tabla 18. CU-S-02, Buscar Vulnerabilidades SQL Injection.

ID	CU-S-03
Nombre	Generar y presentar informe de resultados.
Objetivo	Proporcionar un informe detallado de los resultados que permita a los expertos de seguridad sacar conclusiones sobre las aplicaciones analizadas.
Descripción	El programa sigue los pasos de las dos funciones descritas

[10] Apk es la extensión de los ficheros empaquetados de las aplicaciones Android (Apk Files).

ID	CU-S-03
	arriba, y va generando un informe que es presentado al final de la búsqueda.
Pre-condiciones	Ejecución de alguno de las dos funciones descritas arriba.
Post-condiciones	La aplicación presenta el informe generado.
Escenario	1. El programa va siguiendo los pasos de decompilación y búsqueda. 2. El programa va generando reportes que se añaden al informe. 3. El informe se presenta terminado al usuario del programa.
Causas de fallo	No tener permisos de escritura sobre el directorio de ejecución.

Tabla 19. CU-S-03, Generar y Presentar un Informe de Resultados.

ID	CU-S-04
Nombre	Decompilación y búsqueda automática.
Objetivo	Realizar los pasos anteriores con una sola instrucción en línea de comandos.
Descripción	El usuario llamará al programa por línea de comandos y solo le pasara la dirección de aplicaciones Android, el programa realizará automáticamente las tres funciones anteriores.
Pre-condiciones	El directorio de aplicaciones debe existir y contener aplicaciones.
Post-condiciones	La aplicación presenta el informe.
Escenario	1. El usuario lanza por línea el programa y le pasa solo el directorio con las aplicaciones. 2. La aplicación decompila las aplicaciones del directorio, busca las posibles vulnerabilidades, genera el informe y lo muestra por pantalla.
Causas de fallo	El directorio de aplicaciones está vacío. Los archivos del directorio de aplicaciones no son ficheros apk o compatibles. No tener permisos de escritura sobre el directorio de ejecución.

Tabla 20. CU-S-04, Decompilación y Búsqueda Automática.

2.5. Requisitos

En el apartado que sigue, se presentan los requisitos de software de ambos programas. Estos requisitos se dividirán en dos familias, los requisitos funcionales y los no funcionales.

Los requisitos se presentarán en tablas como está:

ID	
Título	
Descripción	
Prioridad	
Dificultad	
Verificabilidad	
Trazabilidad	

Tabla 21. Tabla de Ejemplo para Requisitos

- **ID:** Código identificativo único con el formato RX-Y-ZZ. La X representa con "F", a los requisitos funcionales, y con "N", a los no funcionales. La Y representa con "A", a la aplicación Android, y con "S", a la aplicación de Búsqueda. Las ZZ representarán un numero correlativo iniciado en 00.
- **Título:** Título descriptivo para el requisito.
- **Descripción:** Breve descripción del requisito.
- **Prioridad:** Necesidad de implementación de este requisito. [Alta-Media-Baja]
- **Dificultad:** Dificultad de implementación de este requisito. [Alta-Media-Baja]
- **Verificabilidad:** Facilidad para verificar este requisito. [Alta-Media-Baja]
- **Trazabilidad**: Trazabilidad a casos de uso.

2.5.1. Requisitos Funcionales – Aplicación Android

ID	RF-A-01
Título	Disponibilidad de ejercicios.
Descripción	El programa dispondrá de los cinco desafíos descritos en esta memoria: Ejercicio de Login, búsqueda de producto, añadir un producto nuevo, y dos ejercicios sobre una aplicación de chat.
Prioridad	☑ Alta ☐ Media ☐ Baja
Dificultad	☐ Alta ☐ Media ☑ Baja
Verificabilidad	☑ Alta ☐ Media ☐ Baja
Trazabilidad	CU-A-01 a CU-A-05

Tabla 22. RF-A-01, Disponibilidad de ejercicios.

ID	RF-A-02
Título	Restablecer la Base de datos.
Descripción	El programa dispondrá de una opción para restablecer la base de datos al estado inicial.
Prioridad	☑ Alta ☐ Media ☐ Baja
Dificultad	☐ Alta ☐ Media ☑ Baja

ID	RF-A-02
Verificabilidad	☑ Alta ☐ Media ☐ Baja
Trazabilidad	CU-A-06

Tabla 23. RF-A-02, Restablecer la Base de datos.

ID	RF-A-03
Título	Ayuda al usuario.
Descripción	El programa dispondrá de una ayuda al usuario, que ira guiando a este hasta la consecución del objetivo.
Prioridad	☐ Alta ☑ Media ☐ Baja
Dificultad	☐ Alta ☑ Media ☐ Baja
Verificabilidad	☐ Alta ☑ Media ☐ Baja
Trazabilidad	CU-A-01 a CU-A-05

Tabla 24. RF-A-03, Ayuda al usuario.

ID	RF-A-04
Título	Consecución de objetivos.
Descripción	El programa notificará al usuario de que ha conseguido superar el desafío.
Prioridad	☑ Alta ☐ Media ☐ Baja
Dificultad	☑ Alta ☐ Media ☐ Baja
Verificabilidad	☑ Alta ☐ Media ☐ Baja
Trazabilidad	CU-A-01 a CU-A-05

Tabla 25. RF-A-04, Consecución de objetivos.

2.5.2. Requisitos Funcionales – Aplicación Shell Script

ID	RF-S-01
Título	Decompilación.
Descripción	El programa dispondrá de una opción individual para decompilar aplicaciones Android.
Prioridad	☑ Alta ☐ Media ☐ Baja
Dificultad	☐ Alta ☐ Media ☑ Baja
Verificabilidad	☑ Alta ☐ Media ☐ Baja
Trazabilidad	CU-S-01

Tabla 26. RF-S-01, Decompilación.

ID	RF-S-02
Título	Búsqueda de vulnerabilidades.
Descripción	El programa dispondrá de una opción individual para buscar posibles vulnerabilidades SQL Injection en el código fuente de las aplicaciones decompiladas.
Prioridad	☑ Alta ☐ Media ☐ Baja
Dificultad	☑ Alta ☐ Media ☐ Baja
Verificabilidad	☑ Alta ☐ Media ☐ Baja
Trazabilidad	CU-S-02

Tabla 27. RF-S-02, Búsqueda de vulnerabilidades.

ID	RF-S-03
Título	Generación de Informes.
Descripción	El programa deberá ser capaz de generar informes.
Prioridad	☑ Alta ☐ Media ☐ Baja
Dificultad	☐ Alta ☐ Media ☑ Baja
Verificabilidad	☑ Alta ☐ Media ☐ Baja
Trazabilidad	CU-S-03

Tabla 28. RF-S-03, Generación de Informes.

ID	RF-S-04
Título	Presentación de Informes.
Descripción	El programa deberá ser capaz de presentar los informes una vez se hayan completado las operaciones anteriores de decompilación y/o búsqueda de vulnerabilidades.
Prioridad	☐ Alta ☐ Media ☑ Baja
Dificultad	☐ Alta ☐ Media ☑ Baja
Verificabilidad	☑ Alta ☐ Media ☐ Baja
Trazabilidad	CU-S-01 a CU-S-04

Tabla 29. RF-S-04, Presentación de Informes.

ID	RF-S-05
Título	Proceso automático.
Descripción	El programa deberá ser capaz de realizar todas las operaciones descritas decompilación, búsqueda de vulnerabilidades, generación de informes, y presentación de informes, de manera automatizada, y mediante el uso de un solo comando.
Prioridad	☐ Alta ☐ Media ☑ Baja

ID	RF-S-05
Dificultad	☐ Alta ☑ Media ☐ Baja
Verificabilidad	☑ Alta ☐ Media ☐ Baja
Trazabilidad	CU-S-04

Tabla 30. RF-S-05, Proceso automático.

2.5.3. Requisitos No Funcionales – Aplicación Android

ID	RN-A-01
Título	Aplicación Android.
Descripción	La aplicación deberá estar implementada en Java para correr en dispositivos Android.
Prioridad	☑ Alta ☐ Media ☐ Baja
Dificultad	☐ Alta ☐ Media ☑ Baja
Verificabilidad	☑ Alta ☐ Media ☐ Baja
Trazabilidad	

Tabla 31. RN-A-01, Aplicación Android.

ID	RN-A-02
Título	Listar desafíos.
Descripción	La aplicación deberá mostrar un menú principal donde se listen las actividades disponibles.
Prioridad	☑ Alta ☐ Media ☐ Baja
Dificultad	☐ Alta ☐ Media ☑ Baja
Verificabilidad	☑ Alta ☐ Media ☐ Baja
Trazabilidad	CU-A-01 a CU-A-05

Tabla 32. RN-A-02, Listar desafíos.

ID	RN-A-03
Título	Menú de opciones.
Descripción	La aplicación deberá presentar un menú de opciones donde se muestre el botón de restablecer base de datos.
Prioridad	☐ Alta ☑ Media ☐ Baja
Dificultad	☐ Alta ☐ Media ☑ Baja
Verificabilidad	☑ Alta ☐ Media ☐ Baja
Trazabilidad	CU-A-06

Tabla 33. RN-A-03, Menú de opciones.

ID	RN-A-04
Título	Orientación de la pantalla.
Descripción	La aplicación deberá tener la pantalla con su orientación de en vertical.
Prioridad	☐ Alta ☐ Media ☑ Baja
Dificultad	☐ Alta ☐ Media ☑ Baja
Verificabilidad	☑ Alta ☐ Media ☐ Baja
Trazabilidad	

Tabla 34. RN-A-04, Orientación de la pantalla.

ID	RN-A-05
Título	Compatibilidad con pestañas.
Descripción	La versión mínima compatible de Android deberá ser la 3.0 para que permita el uso de pestañas (Fragments, 2014).
Prioridad	☑ Alta ☐ Media ☐ Baja
Dificultad	☐ Alta ☐ Media ☑ Baja
Verificabilidad	☑ Alta ☐ Media ☐ Baja
Trazabilidad	

Tabla 35. RN-A-05, Orientación de la pantalla.

2.5.4. Requisitos No Funcionales – Aplicación Shell Script

ID	RN-S-01
Título	Aplicación Shell Script.
Descripción	La aplicación deberá estar implementada Shell Script compatible con GNU-Linux y Mac OS X.
Prioridad	☑ Alta ☐ Media ☐ Baja
Dificultad	☐ Alta ☐ Media ☑ Baja
Verificabilidad	☑ Alta ☐ Media ☐ Baja
Trazabilidad	CU-S-02, CU-S-03

Tabla 36. RN-S-01, Aplicación Shell Script.

ID	RN-S-02
Título	Apktool embebido.
Descripción	La aplicación hará uso de la herramienta Apktool, que deberá estar dentro del directorio de aplicaciones del programa.
Prioridad	☑ Alta ☐ Media ☐ Baja
Dificultad	☐ Alta ☐ Media ☑ Baja
Verificabilidad	☑ Alta ☐ Media ☐ Baja

ID	RN-S-02
Trazabilidad	CU-S-01

Tabla 37. RN-S-02, Apktool embebido.

ID	RN-S-03
Título	Paso de parámetros.
Descripción	La aplicación deberá permitir el paso de uno o dos parámetros como máximo, donde uno de ellos será obligatoriamente la ruta del directorio con aplicaciones Android.
Prioridad	☑ Alta ☐ Media ☐ Baja
Dificultad	☐ Alta ☑ Media ☐ Baja
Verificabilidad	☑ Alta ☐ Media ☐ Baja
Trazabilidad	CU-S-01, CU-S-02, CU-S-04

Tabla 38. RN-S-03, Paso de parámetros.

ID	RN-S-04
Título	Informe detallado.
Descripción	El informe deberá indicar el número de aplicaciones que se procesan, y el nombre de éstas. Además en el apartado de búsqueda de vulnerabilidades, deberán separarse por aplicación, clase de vulnerabilidad, localización de la posible vulnerabilidad, y número total de cada clase.
Prioridad	☐ Alta ☑ Media ☐ Baja
Dificultad	☐ Alta ☑ Media ☐ Baja
Verificabilidad	☑ Alta ☐ Media ☐ Baja
Trazabilidad	CU-S-03

Tabla 39. RN-S-04, Informe detallado.

2.6. Marco Regulador

Las aplicaciones generalmente utilizan datos personales y por ello están sujetas a la Ley Orgánica de protección de Datos (LOPD, 1999).

"La Ley Orgánica 15/1999 de 13 de diciembre de Protección de Datos de Carácter Personal (LOPD) tiene por objeto garantizar y proteger, en lo que concierne al tratamiento de los datos personales, las libertades públicas y los derechos fundamentales de las personas físicas, y especialmente de su honor e intimidad personal y familiar." (LODP Informática, 2014)

Aunque este trabajo no está estrictamente relacionado con lo anterior, sí que sirve para averiguar si estas aplicaciones son conformes a ciertas protecciones que se enuncian en la citada ley, y dado que trata de mostrar debilidades en el diseño, no

sigue las directrices de la ley, y deja desprotegido datos que deberían ser confidenciales. Cabe señalar que todos estos datos son ficticios, y se trata de una aplicación educativa, por lo que en dicho caso, no se estaría incumpliendo la LOPD.

2.7. Restricciones

Si se trata el campo de la seguridad, orientado a los fallos cometidos durante un mal diseño, o programación de aplicaciones, habría muchas posibles vulnerabilidades y errores de programación que tratar en este proyecto. Es por ello que este trabajo solo se centra en los ataques sobre SQL.

2.8. Pruebas

En el apartado de pruebas se enumerarán una serie de pruebas de aceptación, que satisfará todos los requisitos presentados en el apartado con mismo nombre.

Como en ocasiones anteriores, las pruebas se separarán entre las referentes a la aplicación Android y a la aplicación de búsqueda de vulnerabilidades.

Cada una de las pruebas irá documentada en una tabla cómo la siguiente:

ID	
Trazabilidad	
Descripción	

Tabla 40. Tabla de Ejemplo para pruebas de aceptación.

- **ID:** Código identificativo único con el formato PR-X-YY. La X representa con "A", a la aplicación Android, y con "S", a la aplicación de Búsqueda. Las YY representarán un número correlativo iniciado en 00.
- **Trazabilidad:** Trazabilidad a requisitos.
- **Descripción:** Explicación de las pruebas realizadas para verificar el cumplimiento de los requisitos.

2.8.1. Catalogo de Pruebas – Aplicación Android

ID	PR-A-01
Trazabilidad	RF-A-01, RN-A-01, RN-A-02, RN-A-04, RN-A-05
Descripción	Se abrirá la aplicación, y se verá que puede accederse a cada una de las actividades ofrecidas en el programa: ejercicio de login, búsqueda de producto, incorporación de un nuevo producto, chat 1, y chat2.

Tabla 41. PR-A-01.

ID	PR-A-02
Trazabilidad	RF-A-01, RN-A-01 a RN-A-04
Descripción	Se abrirá la aplicación, y se pulsara el botón de opciones para que se despliegue el botón de restablecer base de datos.

Tabla 42. PR-A-02.

ID	PR-A-03
Trazabilidad	RF-A-01, RF-A-02, RN-A-01 a RN-A-05
Descripción	Se abrirá el ejercicio de Chat 1 y se mandará un mensaje. Se volverá al menú principal y se pulsará el botón de restablecer base de datos. Se volverá a entrar en el ejercicio de Chat 1, y se verá que el mensaje ha desaparecido.

Tabla 43. PR-A-03.

ID	PR-A-04
Trazabilidad	RF-A-01, RF-A-03, RN-A-01
Descripción	Se introducirá una comilla simple ‘ en cualquiera de los campos de cualquier ejercicio y se dará a enviar o al botón correspondiente hasta que salte la primera ayuda.

Tabla 44. PR-A-04.

ID	PR-A-05
Trazabilidad	RF-A-01, RF-A-04, RN-A-01
Descripción	Se superará cualquiera de los desafíos, y podrá verse una notificación reflejando que el sistema ha sido vulnerado.

Tabla 45. PR-A-05.

2.8.2. Catalogo de Pruebas – Aplicación Shell Script

ID	PR-S-01
Trazabilidad	RF-S-01, RN-S-01, RN-S-02
Descripción	Se lanzará la aplicación con la opción –d, indicando la ruta de un directorio que contenga alguna aplicación Android con extensión apk. Terminada la ejecución del programa se consultará en el directorio “dapks” comprobando la existencia de las aplicaciones Android decompiladas.

Tabla 46. PR-S-01.

ID	PR-S-02
Trazabilidad	RF-S-02, RN-S-01
Descripción	Se lanzará la aplicación con la opción –s, indicando la ruta de un directorio que contenga alguna aplicación Android decompilada. Terminada la ejecución del programa se verificará el resultado del informe mostrado por pantalla.

Tabla 47. PR-S-02.

ID	PR-S-03
Trazabilidad	RF-S-01, RF-S-02, RF-S-03, RF-S-05, RN-S-01, RN-S-02, RN-S-04
Descripción	Se lanzará la aplicación con cualquiera de sus opciones, y se comprobará que en el directorio de ejecución se ha creado un informe con título: injectSearch_report_[fecha]_[hora].txt

Tabla 48. PR-S-03.

ID	PR-S-04
Trazabilidad	RF-S-01 a RF-S-05, RF-S-03, RN-S-01, RN-S-02, RN-S-04
Descripción	Se lanzará la aplicación con cualquiera de sus opciones –d, –s, o simplemente con el directorio, y se comprobará que se muestra por pantalla el informe generado.

Tabla 49. PR-S-04.

ID	PR-S-05
Trazabilidad	RF-S-05, RN-S-01, RN-S-02, RN-S-04
Descripción	Se lanzará la aplicación indicando únicamente la ruta de un directorio que contenga alguna aplicación Android con extensión apk. Terminada la ejecución del programa se consultará el resultado del informe por pantalla.

Tabla 50. PR-S-05.

ID	PR-S-06
Trazabilidad	RN-S-01, RN-S-03
Descripción	Se lanzará la aplicación sin parámetros y se comprobará que se muestra una ayuda de cómo debe utilizarse el programa. Se lanzará la aplicación con tres parámetros y se verá que nos muestra una ayuda de cómo debe utilizarse el programa. Se lanzará la aplicación indicando la ruta de una carpeta con aplicaciones Android en formato apk, y se verá que el programa se ejecuta sin problemas. Se lanzará la aplicación indicando la opción –d o –s, y la ruta de una carpeta con aplicaciones Android en formato apk, o aplicaciones Android decompiladas, y se verá que el programa se ejecuta sin problemas

Tabla 51. PR-S-06.

2.8.3. Matriz de trazabilidad Requisitos – Pruebas de Aceptación Android

	PR-A-01	PR-A-02	PR-A-03	PR-A-04	PR-A-05
RF-A-01	x	x	x	x	x
RF-A-02		x	x		
RF-A-03				x	
RF-A-04					x
RN-A-01	x	x	x	x	x
RN-A-02	x	x	x		
RN-A-03		x	x		
RN-A-04	x	x	x		
RN-A-05	x		x		

Tabla 52. Matriz de trazabilidad Requisitos – Pruebas de Aceptación Android.

2.8.4. Matriz de trazabilidad Requisitos – Pruebas de Aceptación Shell Script

	PR-S-01	PR-S-02	PR-S-03	PR-S-04	PR-S-05	PR-S-06
RF-S-01	x		x	x		
RF-S-02		x	x	x		
RF-S-03			x	x		
RF-S-04				x		
RF-S-05			x	x	x	
RN-S-01	x	x	x	x	x	x
RN-S-02	x		x	x	x	
RN-S-03						x
RN-S-04			x	x	x	

Tabla 53. Matriz de trazabilidad Requisitos – Pruebas de Aceptación Shell Script.

2.9. Diagrama de flujo de Shell Script

Para que se comprenda mejor el funcionamiento de la aplicación en Shell Script, se ha realizado es siguiente diagrama de flujo, representado en la Figura 14.

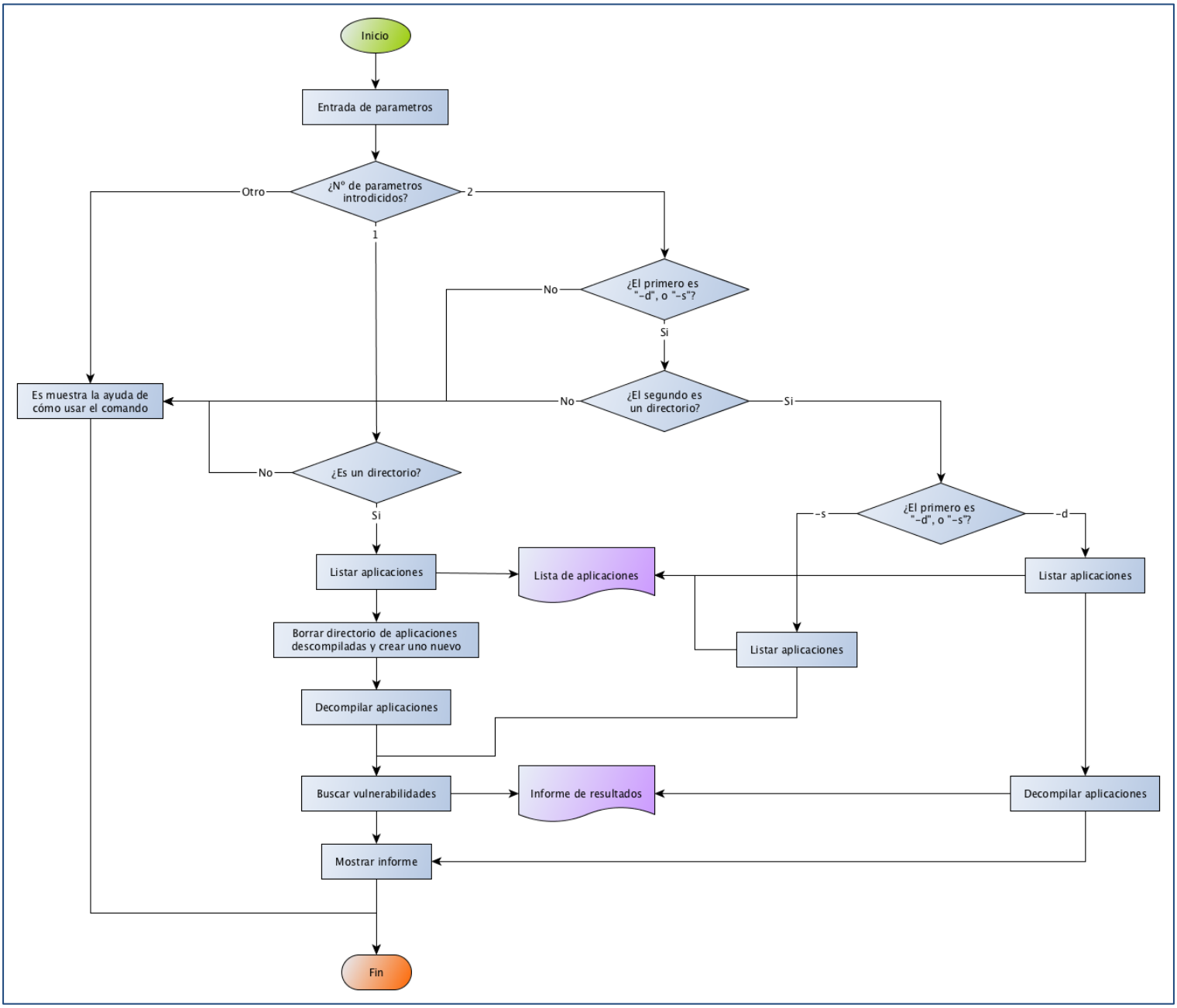

Figura 14. Diagrama de Flujo General del Script.

Capítulo 3. Diseño

En este capitulo se detalla el proceso de diseño de las dos aplicaciones. Estudiando los componentes que forman cada una de ellas, y el patrón de diseño en el que se basan.

3.1. Diseño de la arquitectura

La primera de las aplicaciones, es un programa desarrollado para Android, por los que sigue el patrón propuesto de Modelo-Vista-Controlador (MVC Android, 2014).

El patrón de arquitectura de software MVC, consiste en separar en tres componentes distintos el modelo, la vista y el controlador. El modelo es el encargado de trabajar con la información almacenada que necesita utilizar la aplicación. La vista es la interfaz con la que los usuarios interactúan y desde la que visualizan la información o solicitan operaciones al programa. El controlador es la parte del programa que contiene la lógica de funcionamiento, y que interactúa con las otras dos partes del patrón, el modelo, y la vista, dando vida a la aplicación.

Se puede entender, de manera gráfica, el patrón de Modelo-Vista-Controlado, observando la Figura 15.

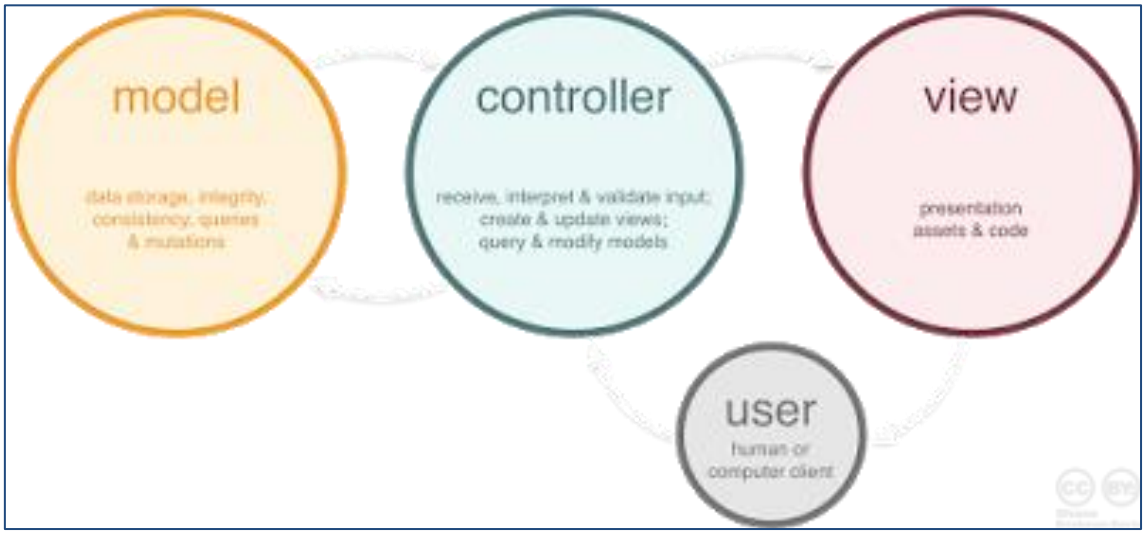

Figura 15. MVC en Android[11].

En lo referente a la segunda de las aplicaciones, al tratarse de un Shell Script, no ha seguido el patrón de diseño presentado arriba. El Script se ha implementado en un solo fichero con un listado de operaciones colocadas en el orden de su ejecución, y haciendo uso de funciones para mantener el orden en su estructura.

[11] Imagen obtenida en (MVC Android 2).

3.2. Diagrama de componentes

Para la aplicación Android el diagrama de componentes desarrollado ha sido el representado en la Figura 16:

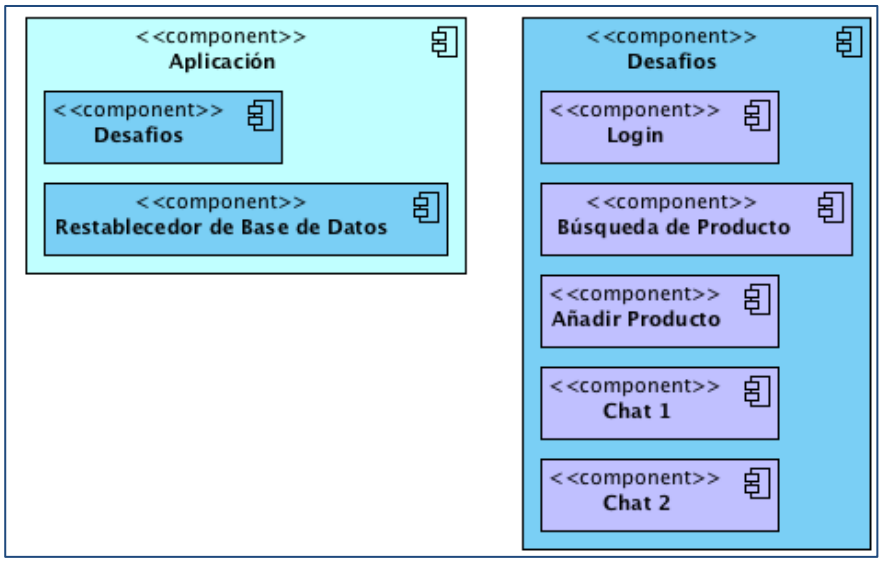

Figura 16. Diagrama de Componentes en la Aplicación Android.

Para el Script de análisis de vulnerabilidades ante SQL Injection, el diagrama de componentes desarrollado se ha representado de la siguiente manera, en la Figura 17:

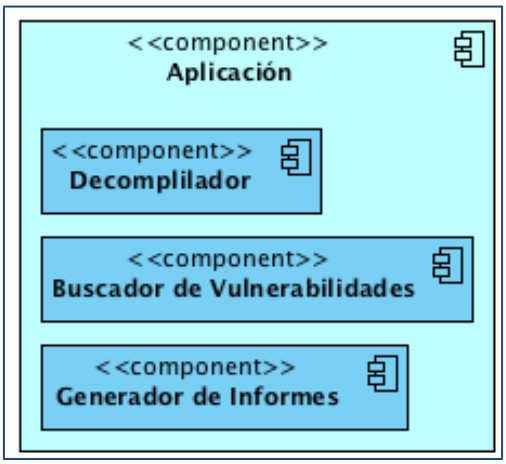

Figura 17. Diagrama de Componentes en el Script.

3.3. Modelo de datos

Dada la simplicidad de la aplicación encargada de buscar vulnerabilidades en las aplicaciones Android, no ha sido necesario establecer un modelo de datos para ella, sin embargo, la aplicación Android trata precisamente de la vulnerabilidad de las bases de datos, a causa de una mala programación. La aplicación móvil por tanto presenta un modelos de datos, que sin poca relación entre sus elementos, forma un modelo de datos válido.

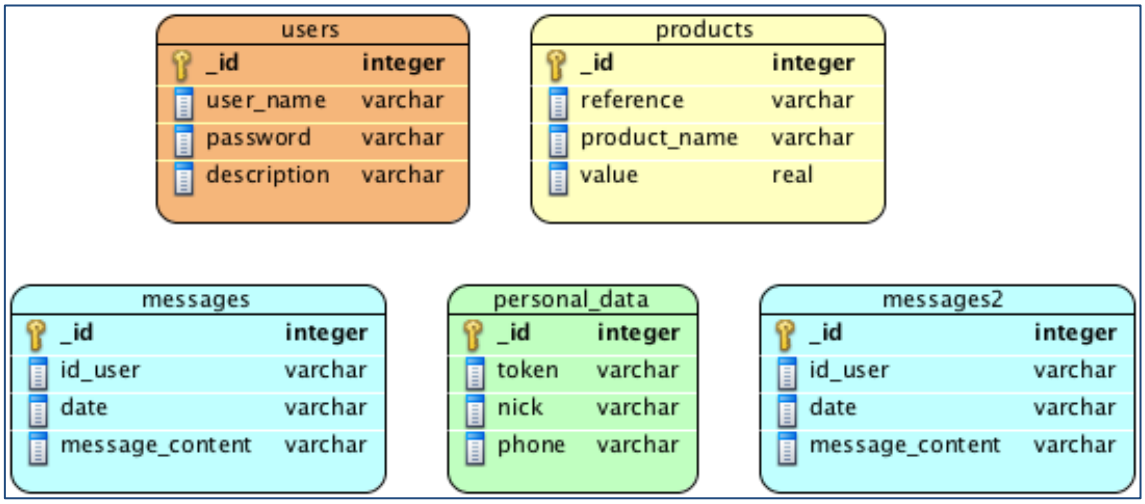

Figura 18. Modelo de Datos Aplicación en Android.

Al tratarse de ejercicios con temáticas distintas, las tablas del modelo (Figura 18) no se encuentran interconectadas, y pese que se sabe que mediante ataques SQL Injection es posible acceder a cualquiera, el uso asignado de cada una de las tablas, a cada ejercicio es el siguiente:

- Ejercicio Login → Tabla users
- Ejercicio Búsqueda de producto → Tabla products
- Ejercicio Añadir producto → Tabla products
- Ejercicio Chat 1 → Tablas Messages y personal_data*
- Ejercicio Chat 2 → Tablas Messages2 y personal_data*

*) personal_data es una tabla auxiliar de la aplicación de chat que no debería ser accesible desde la ventana de conversación, los datos que almacena son los correspondientes a la cuenta del usuario de la aplicación de chat, y que le identifican de forma única en el servidor de la compañía.

3.4. Desglose de clases y funciones principales

Presentados el diseño de la arquitectura, los diagramas de componentes, y el modelo de datos de las aplicaciones, más arriba, se pasará a desglosar la aplicación en sus clases y funciones principales, en según su pertenencia a cada una de las aplicaciones, y a su localización en el modelo o en el controlador.

3.4.1. Aplicación Android

En la sección del modelo podemos encontrar las siguientes clases:

- AppStorageHelper
- SQLinjectStorageManager

AppStorageHelper	
Atributos	
DATABASE_VERSION	Versión de la base de datos.
DATABASE_NAME	Nombre de la base de datos.

context	Contexto de la aplicación.
Métodos	
onCreate	Método que crea la base de datos, y la puebla con la información inicial.
fillInDatabase	Método que puebla la base de datos con la información inicial.
resetDatabase	Método que borra el contenido de la base de datos y la puebla con la información inicial.
checkDataBase	Método que retorna la existencia o no de la base de datos pasada como parámetro, creando la instancia si no existe.

Tabla 54. AppStorageHelper.

SQLinjectStorageManager	
Atributos	
TOAST_TIME	Tiempo que se mostrarán las notificaciones "toast".
Métodos	
checkDbExistis	Método que retorna la existencia o no de la base de datos pasada como parámetro.
set_product	Método que inserta un nuevo producto en la base de datos.
set_message	Método que inserta un nuevo mensaje en la base de datos.
get_UserLine	Método que retorna de la base de datos los _id, user_name y description, del usuario pasado por parámetro.
get_AdminPassword	Método que retorna de la base de datos, la contraseña del usuario administrador.
get_Product	Método que retorna de la base de datos los reference, product_name y value, del producto pasado por parámetro.
get_NumProduct	Método que retorna el número de productos almacenados en la base de datos.
get_messages	Método que retorna los mensajes almacenados en la base de datos.
readCursor	Método que formatea la salida de las consultas a la base de datos.

Tabla 55. SQLinjectStorageManager.

En la sección del controlador cuenta con las clases:

- MainActivity
- Chat1Activity
- Chat2Activity
- ChatExplanationActivity
- ContactFragmentChat1
- ContactFragmentChat2
- EnlistProductActivity
- FormatearFecha
- HelpActivity

- LoginActivity
- OneComment
- SearchProductActivity
- SecureAreaActivity
- ShowProductsActivity

MainActivity	
Atributos	
challengesLV	Listview donde se cargará la lista de desafíos.
Métodos	
onCreate	Método que comprueba la existencia de la base de datos, y lista los desafíos de la aplicación, proporcionando el enlace a estos si se pulsa sobre ellos.
onCreateOptionsMenu	Método que despliega el botón de "Restablecer base de datos".
onOptionsItemSelected	Método que lanza la ejecución del botón del menú que ha sido pulsado, en este caso "Restablecer base de datos".

Tabla 56. MainActivity.

Chat1Activity	
Atributos	
SCREEN_WIDTH	Variable que almacena el ancho de pantalla.
helpCounter1	Contador de errores para mostrar la ayuda 1.
helpCounter2	Contador de errores para mostrar la ayuda 2.
helpCounter3	Contador de errores para mostrar la ayuda 3.
Métodos	
onCreate	Método que muestra el objetivo del ejercicio, y carga las pestañas de las conversaciones.
onTabSelected	Método que detectada la pulsación de las pestañas, carga el "fragment" asociado al ejercicio de Chat 1.
showExplanation	Método que muestra en una ventana emergente el objetivo del ejercicio, cargando la clase ChatExplanationActivity.

Tabla 57. Chat1Activity.

Chat2Activity	
Atributos	
SCREEN_WIDTH	Variable que almacena el ancho de pantalla.
helpCounter1	Contador de errores para mostrar la ayuda 1.
helpCounter2	Contador de errores para mostrar la ayuda 2.
helpCounter3	Contador de errores para mostrar la ayuda 3.
Métodos	
onCreate	Método que muestra el objetivo del ejercicio, y carga las pestañas de las conversaciones.
onTabSelected	Método que detectada la pulsación de las pestañas,

	carga el “fragment” asociado al ejercicio de Chat 2.
showExplanation	Método que muestra en una ventana emergente el objetivo del ejercicio, cargando la clase ChatExplanationActivity.

Tabla 58. Chat2Activity.

ChatExplanationActivity	
Atributos	
Métodos	
onCreate	Método que muestra el objetivo asociado al ejercicio de chat, en función del ejercicio que ha cargado esta clase.

Tabla 59. ChatExplanationActivity.

ContactFragmentChat1	
Atributos	
yourChat	Variable booleana que indica si estás visualizando tu chat, o el chat del interlocutor.
myView	Variable que identifica la vista.
context	Contexto de la aplicación.
maxPixWidth	Valor máximo de pixeles para definir el ancho de la pantalla.
adapter	Objeto “adapter” donde se cargan los mensajes de las conversaciones.
textoMsg	Campo de texto donde se escriben los mensajes a enviar.
botonEnviarMensaje	Botón de enviar mensaje.
chatView	Objeto donde se cargan los mensajes del “adapter”.
Métodos	
setItsYourChat	Método que define que ventana de chat se está visualizando, la del usuario, o la del interlocutor.
onResume	Método que lanza las funciones que cargan los mensajes enviados en pantalla, y que envía los mensajes al servidor cuando se pulsa al botón de enviar.
fillInAdapter	Método que carga en el “adapter”, los mensajes almacenados en la base de datos, y que están relacionado con dicha conversación. Este método también es el encargado de detectar si se ha conseguido superar el objetivo del ejercicio de chat 1, notificando al usuario de tal logro.
showHelp	Método cuya tarea es cargar las ayudas de usuario.

Tabla 60. ContactFragmentChat1.

Dentro de la clase ContactFragmentChat1, encontramos una clase embebida llamada: ChatArrayAdapter, que se encarga de formatear los elementos del "adapter" que será empleado en la clase arriba definida.

ContactFragmentChat2	
Atributos	
yourChat	Variable booleana que indica si estas visualizando tu chat, o el chat del interlocutor.
myView	Variable que identifica la vista.
context	Contexto de la aplicación.
maxPixWidth	Valor máximo de pixeles para definir el ancho de la pantalla.
adapter	Objeto "adapter" donde se cargan los mensajes de las conversaciones.
textoMsg	Campo de texto donde se escriben los mensajes a enviar.
botonEnviarMensaje	Botón de enviar mensaje.
chatView	Objeto donde se cargan los mensajes del "adapter".
Métodos	
setItsYourChat	Método que define que ventana de chat se está visualizando, la del usuario, o la del interlocutor.
onResume	Método que lanza las funciones que cargan los mensajes enviados en pantalla, y que envía los mensajes al servidor cuando se pulsa al botón de enviar.
fillInAdapter	Método que carga en el "adapter", los mensajes almacenados en la base de datos, y que están relacionado con dicha conversación. Este método también es el encargado de detectar si se ha conseguido superar el objetivo del ejercicio de chat 2, notificando al usuario de tal logro.
showHelp	Método cuya tarea es cargar las ayudas de usuario.

Tabla 61. ContactFragmentChat2.

Dentro de la clase ContactFragmentChat2, encontramos una clase embebida llamada: ChatArrayAdapter, que se encarga de formatear los elementos del "adapter" que será empleado en la clase arriba definida.

EnlistProductActivity	
Atributos	
helpCounter1	Contador de errores para mostrar la ayuda 1.
helpCounter2	Contador de errores para mostrar la ayuda 2.
Métodos	
logicaBtnEnlistProductSave	Método que manda comprobar que todos los elementos del formulario estén cumplimentados, guarda en la base de datos el nuevo producto, y lanza la clase que mostrará el resultado de la

	operación recién hecha.
showHelp	Método cuya tarea es cargar las ayudas de usuario.
isNotEmptyBox	Método que verifica si el campo indicado por parámetro está cumplimentado.

Tabla 62. EnlistProductActivity.

FormatearFecha	
Atributos	
mes	Array que contienen todos los meses del año en formato texto.
Métodos	
formatoUno	Método que formatea la fecha en un formato “dd de <mes> de yyyy".

Tabla 63. FormatearFecha.

HelpActivity	
Atributos	
Métodos	
onCreate	Método que en función del identificador recibido, muestra por pantalla una ventana con la ayuda correspondiente al identificador.

Tabla 64. HelpActivity.

LoginActivity	
Atributos	
helpCounter1	Contador de errores para mostrar la ayuda 1.
helpCounter2	Contador de errores para mostrar la ayuda 2.
helpCounter3	Contador de errores para mostrar la ayuda 3.
Métodos	
logicaBtnLoginSend	Método que manda comprobar que todos los elementos del formulario estén cumplimentados, y verifica que los datos de usuario y contraseña introducidos corresponden con los almacenados en la base de datos.
showHelp	Método cuya tarea es cargar las ayudas de usuario.
isNotEmptyBox	Método que verifica si el campo indicado por parámetro está cumplimentado.

Tabla 65. LoginActivity.

OneComment	
Atributos	
izquierda	Variable que indica si el comentario debe colocarse a la izquierda o a la derecha de la pantalla de chat.

comentario	Contenido del comentario.
hora	Hora de envío del comentario.
Métodos	
Getters y Setters	Métodos para rellenar y extraer la información de los atributos.

Tabla 66. OneComment.

SearchProductActivity	
Atributos	
helpCounter1	Contador de errores para mostrar la ayuda 1.
helpCounter2	Contador de errores para mostrar la ayuda 2.
helpCounter3	Contador de errores para mostrar la ayuda 3.
Métodos	
logicaBtnSearch	Método que busca en la base de datos, el producto indicado, y lanza la visualización del resultado de dicha búsqueda.
showHelp	Método cuya tarea es cargar las ayudas de usuario.

Tabla 67. SearchProductActivity.

SecureAreaActivity	
Atributos	
Métodos	
onCreate	Método que visualiza el área restringida tras la realización de un login "valido".

Tabla 68. SecureAreaActivity.

ShowProductsActivity	
Atributos	
Métodos	
onCreate	Método que lanza la carga de productos de la base de datos que coincidan con los parámetros introducidos en el campo de búsqueda de SearchProductActivity.
showHelp	Método cuya tarea es cargar las ayudas de usuario.

Tabla 69. ShowProductsActivity.

3.4.2. Aplicación Shell Script

La aplicación de Shell Script a diferencia de la aplicación Android, está autocontenida en un único fichero, por lo que no está descompuesto en clases. Que no se encuentre organizada en clases no quiere decir que su lógica esté desorganizada.

Las clases que podemos encontrar en el Script son las siguientes:

- fun_list
- fun_decompile
- fun_search

Shell Script	
fun_list	Esta función se encarga de recorrer el listado de aplicaciones y generar una lista que luego será utilizada por las otras dos funciones a continuación. Como función secundaria, lo que hace es inicializar el informe que irá siendo rellenado con el resto de pasos.
fun_decompile	Esta función se encarga de recorrer el listado generado en con la función anterior e ir decompilando cada una de las aplicaciones Android enumeradas en la lista. A medida que se realizan las operaciones de decompilación, la función va añadiendo trazas al informe de resultados.
fun_search	Esta función se encarga de recorrer el listado generado en la función fun_list(), e ir buscando en cada una de las aplicaciones decompiladas si hay una posible vulnerabilidad frente a ataques SQL Injection a causa de una mala programación. Como también lo hacia la función anterior, a medida que se van realizando operaciones de búsqueda, se van añadiendo trazas al informe de resultados.

Tabla 70. Funciones del Shell Script.

Capítulo 4. Vulnerabilidades añadidas

La aplicación Android presentada, ha sido deliberadamente desarrollada con errores de programación que desembocan en graves fallos de seguridad. La finalidad de cometer estos errores es concienciar a los desarrolladores de aplicaciones para dispositivos móviles, de la sensibilidad de la información que estos aparatos almacenan, y de lo perjudicial que puede resultar tanto para los usuarios, como para la empresa desarrolladora o propietaria del software, la revelación de estas informaciones.

La aplicación representa una herramienta interactiva de aprendizaje, divida en cinco ejercicios que enseñan mediante técnicas de SQL injection, otras cinco vulnerabilidades distintas que pueden sufrir las aplicaciones móviles en este caso concreto, pero por extensión a cualquier tipo de plataforma.

En este capítulo se presentarán las vulnerabilidades individuales de cada ejercicio, y se expondrán distintas contramedidas para solventar dichos problemas.

4.1. Ejercicio 1: Login

En el ejercicio 1 la vulnerabilidad que se trata de explotar es la de conseguir entrar en un área restringida sin conocer los datos de acceso a esta.

4.1.1. Motivos de la vulnerabilidad

El principal motivo de la vulnerabilidad viene dado por un mal uso de las funciones de consulta a la base de datos SQLite (rawQuery in Android Developer Web, 2014). Estos métodos que proporciona el API de Android se usan de manera incorrecta al construir la consulta SQL mediante un String al que se le pasan mediante variables, los valores de la consulta.

En el caso de este ejercicio:

```
String sqlQuery = "SELECT _id, user_name, description FROM users WHERE
user_name='"+userName+"' AND password='"+userPass+"'";
```

El segundo error de programación cometido, se trata de la no encapsulación de las cadenas que se almacenan en estas variables. Este fallo le permite a un atacante introducir caracteres problemáticos como la comilla simple ‘ , que puede provocar un funcionamiento irregular en la consulta.

De esta manera nada impediría a un atacante emplear técnicas de SQL injection, y vulnerar la seguridad de esta aplicación.

En el caso de este ejercicio 1, se conseguiría entrar en el área restringida tan solo con poner como usuario la cadena ‘ or 1==1 --.

Esta cadena produciría una consulta como la siguiente:

```
"SELECT _id, user_name, description FROM users WHERE user_name='' or
1=1 -- ' AND password='"+userPass+"'"
```

Teniendo en cuenta que los dos guiones -- simbolizan que lo que sigue es un comentario, la consulta que se estaría produciendo sería: Selecciona los datos de usuario cuando el usuario sea =‘’ o si 1 es igual a 1.

Esto hace que retorne toda la tabla de usuarios ya que la segunda clausula se va a cumplir siempre.

4.1.2. Contramedidas a la vulnerabilidad

Para corregir este tipo de vulnerabilidades, se recomienda revisar los siguientes puntos:

- Hacer una verificación previa del contenido de los campos de texto para evitar caracteres problemáticos.
- Encapsular el contenido de dichos campos de texto.
- Utilizar de forma correcta las funciones del API de Android.

Aunque no protegen directamente del SQL Injection, los siguientes consejos también favorecen a que la aplicación sea más segura:

- Cifrar el contenido de la base de datos basándose en métodos de KeyChain (KeyChain in Android Developer Web, 2014), o Password Based Encryption (PBE in Android Developer Web, 2014).
- No emplear nombres de usuario que sugieran los privilegios de estos, véase “admin” para un usuario administrador.

4.2. Ejercicio 2: Búsqueda de producto

En el ejercicio 2 la vulnerabilidad que se trata de explotar es la de conseguir obtener información de otra tabla a la que originariamente está asociada la consulta.

4.2.1. Motivos de la vulnerabilidad

El principal motivo de la vulnerabilidad vuelve a ser un mal uso de las funciones de consulta a la base de datos SQLite. Estos métodos que proporciona el API de Android se usan de manera incorrecta al construir la consulta SQL mediante un String al que se le pasan mediante variables, los valores de la consulta.

En el caso de este ejercicio:

```
String sqlQuery = "SELECT reference, product_name, value FROM products
WHERE reference LIKE '%"+reference+"%'";
```

El segundo error de programación cometido, como pasa en el caso anterior, se trata de la no encapsulación de las cadenas que se almacenan en estas variables. Este fallo le permite a un atacante introducir caracteres problemáticos como la comilla simple ‘ , que puede provocar un funcionamiento irregular en la consulta.

De esta manera nada impediría a un atacante emplear técnicas de SQL injection, y vulnerar la seguridad de esta aplicación.

De lo que se trata en esta ocasión es de inyectar la clausula UNION para realizar otra consulta interna a la base de datos.

De esta manera podría escribirse en el campo de búsqueda: 'UNION SELECT 'A','B',(another Select query) -- Con lo que se conseguiría añadir una línea de resultados nueva en la que se mostraría el resultado de la consulta inyectada.

Para este ejercicio en concreto lo que se consulta es la tabla de usuarios para obtener la contraseña del usuario administrador.

4.2.2. Contramedidas a la vulnerabilidad

Para corregir este tipo de vulnerabilidades, se recomienda revisar los siguientes puntos:

- Hacer una verificación previa del contenido de los campos de texto para evitar caracteres problemáticos.
- Encapsular el contenido de dichos campos de texto.
- Utilizar de forma correcta las funciones del API de Android.
- Tener en una base de datos distinta las tablas de las cuentas de usuario.
- Revisar si es estrictamente necesario almacenar la contraseña de un administrador.

4.3. Ejercicio 3: Alta de producto

En el ejercicio 3 la vulnerabilidad que se trata de explotar es la de conseguir insertar productos fuera de control.

4.3.1. Motivos de la vulnerabilidad

Lo que se trata de explotar en esta ocasión es una consulta de inserción. El principal motivo de la vulnerabilidad vuelve a ser el mismo que los casos anteriores, un mal uso de las funciones del API de Android, categorizado para instrucciones INSERT (SQLite Insert in Android Developer Web, 2014). Estos métodos que proporciona el API de Android se usan de manera incorrecta al construir la consulta SQL mediante un String al que se le pasan mediante variables, los valores de la inserción.

En el caso de este ejercicio:

```
String sql = "INSERT INTO products(product_name,reference,value)" +
"VALUES ('"+productName+"','"+reference+"',"+price+")";
```

El segundo error de programación cometido, como pasa en los casos anteriores, y en resto de ejercicios, se trata de la no encapsulación de las cadenas que se almacenan en estas variables. Este fallo le permite a un atacante introducir caracteres problemáticos como la comilla simple ‘ , que puede provocar un funcionamiento irregular en la consulta.

De esta manera nada impediría a un atacante emplear técnicas de SQL injection, y vulnerar la seguridad de esta aplicación.

Esta vez lo único que se realiza es modificar la query para que en vez de realizar una inserción simple se realice una instrucción de inserción múltiple. Para que esto ocurra basta con escribir en el primer campo de texto: product_nameA','refA',100),('product_nameB','refB',50)-- consiguiendo de esta manera insertar dos productos que no pasarían control alguno.

4.3.2. Contramedidas a la vulnerabilidad

Para corregir este tipo de vulnerabilidades, se recomienda tener precaución con los mismos puntos que en apartados anteriores:

- Verificación previa del contenido de los campos de texto para evitar caracteres problemáticos.
- Encapsular el contenido de dichos campos de texto.
- Utilizar de forma correcta las funciones del API de Android.

4.4. Ejercicio 4: Chat 1

En el ejercicio 4 la vulnerabilidad que se trata de explotar es la de obtener información de una tabla a la que no se debería tener acceso. Dado que se trata de una instrucción INSERT, lo que se realiza es una consulta que rellena uno de los campos de la instrucción INSERT.

Esta actividad, trata de ejemplificar una vulnerabilidad crítica en sistemas de mensajería: La obtención del token de usuario, un código identificativo que permitiría suplantar la identidad de la víctima, si éste se introdujera en otro dispositivo.

4.4.1. Motivos de la vulnerabilidad

Los motivos de la vulnerabilidad son los mismos que los presentados para el ejercicio 3 en el apartado 4.3.1.

La consulta vulnerable para este ejercicio es:

```
sql="INSERT INTO messages (message_content,id_user,date) " +
"VALUES ('"+msgContent+"','"+phone+"','"+date+"')";
```

En esta instrucción lo que se trata de hacer es sustituir el contenido del mensaje, para que este tome el valor resultante de una consulta en la base de datos. La modificación necesaria para conseguir esto sería: '||(consulta SELECT que devuelva un único resultado)||' donde la instrucción SELECT debería apuntar al elemento que se desease conocer.

4.4.2. Contramedidas a la vulnerabilidad

Para corregir este tipo de vulnerabilidades, se recomienda tener precaución con los mismos puntos que en apartados anteriores, y alguno adicional:

- Verificación previa del contenido de los campos de texto para evitar caracteres problemáticos.
- Encapsular el contenido de dichos campos de texto.

- Utilizar de forma correcta las funciones del API de Android.
- Almacenar el token y otra información de similar importancia en una base de datos distinta.

Aunque no protegen directamente del SQL Injection, el siguiente consejo también favorece a que la aplicación sea más segura:

- Cifrar el contenido de la base de datos basándose en métodos de KeyChain (KeyChain in Android Developer Web, 2014), o Password Based Encryption (PBE in Android Developer Web, 2014). Principalmente la que almacene el token.

4.5. Ejercicio 5: Chat 2

En el ejercicio 5 la vulnerabilidad que se trata de explotar es el mal uso de la función exeSQL (exeSQL in Android Developer Web, 2014), para realizar un ataque de suplantación de identidad en el que se permite engañar a la aplicación de chat para que piense que un mensaje ha sido enviado por la victima.

4.5.1. Motivos de la vulnerabilidad

El motivo de la vulnerabilidad es el mismo descrito que en el apartado anterior, con la particularidad de que lo que se trata de modificar a parte del mensaje, es el identificador de usuario del interlocutor.

La consulta vulnerable de este ejercicio es:

```
sql="INSERT INTO messages2 (message_content,id_user,date) " +
"VALUES ('"+msgContent+"','"+phone+"','"+date+"')";
```

Como se ha dicho arriba, lo que se trata de modificar es el contenido del mensaje, y sobre todo, el identificador de usuario de nuestro conferenciante. Para realizar este cambio, el mensaje que se debería escribir en el campo de texto sería algo parecido a esto: this is a message','+34123123123&1',datetime('now')) --" donde debería sustituirse el mensaje que se quiera enviar, y el teléfono de la victima acabado en &1, que indica que el mensaje ha sido enviado por dicho usuario.

4.5.2. Contramedidas a la vulnerabilidad

Para corregir este tipo de vulnerabilidades, se recomienda tener precaución con los mismos puntos que en apartados anteriores, y alguno adicional:

- Verificación previa del contenido de los campos de texto para evitar caracteres problemáticos.
- Encapsular el contenido de dichos campos de texto.
- Utilizar de forma correcta las funciones del API de Android.

Capítulo 5. Implementación

En este capítulo del proyecto se hará hincapié en los temas relacionado con la implementación de las aplicaciones. El capitulo se dividirá en dos secciones principales. La primera presentará detalles de implementación de las funciones más relevantes, y la segunda ofrecerá el resultado del plan de pruebas elaborado en el capítulo de análisis.

5.1. Aspectos de la implementación

En este apartado se presentarán los detalles relevantes a las funciones más importantes de las dos aplicaciones.

5.1.1. Implementación en aplicación Android

- **Menú principal de la aplicación**

En la implementación del menú principal se puede destacar la programación del listado de ejercicios, y la funcionalidad de restablecimiento de la base de datos.

```
protected void onCreate(Bundle savedInstanceState) {
. . . .

// Check the existence of the database
SQLinjectStorageManager.checkDbExistis(getApplicationContext(),
"SQLinjectDB");

//    ListView for exercises
challengesLV = (ListView) findViewById(R.id.lv_main_challenges);

// ArrayList with the exercises
ArrayList<String> challengeArrayList = new ArrayList<String>();
challengeArrayList.add(0, "Login");
challengeArrayList.add(1, "Search product");
challengeArrayList.add(2, "Enlist product");
challengeArrayList.add(3, "Chat 1");
challengeArrayList.add(4, "Chat 2");

// The adapter loads the data from ArrayList.
final ArrayAdapter<String> arrayAdapter = new
ArrayAdapter<String>(this, android.R.layout.simple_list_item_1,
challengeArrayList );

//    The ListView sets the adapter
challengesLV.setAdapter(arrayAdapter);

. . .
}
```

En las primeras líneas de la aplicación, se observa que la primera acción que se realiza, es verificar la existencia de la base de datos. Esta verificación encadena su creación si no existiera previamente.

El siguiente paso en realizar es cargar el "ListView" del interfaz, en el que se colocará la lista de ejercicios. En los pasos a continuación, se crea el listado, y se rellena el "adapter" que muestra el listado de ejercicios en el interfaz.

```
// Click over Challenge
challengesLV.setOnItemClickListener(new OnItemClickListener() {
. . .

switch(position){
      case 0:
            newIntent = new Intent (getApplicationContext(),
      LoginActivity.class);
            Log.d("SQLinject Debug", "MainActivity: 'Login'
Challenge Accepted!!");
            MainActivity.this.startActivity(newIntent);
            break;
      case 1:
            newIntent = new Intent(MainActivity.this,
SearchProductActivity.class);
            Log.d("SQLinject Debug", "MainActivity: 'Search product'
Challenge Accepted!!");
            MainActivity.this.startActivity(newIntent);
            break;
      . . .
}

. . .
}
```

Para que al pulsar sobre los elementos de la lista la aplicación abra sus ventanas correspondientes, se programan unos "onItemClickListener", con los enlaces a las clases correspondientes de cada desafío.

```
public boolean onOptionsItemSelected(MenuItem item) {
. . .

AppStorageHelper dbHelper = new
AppStorageHelper(this.getApplicationContext());

SQLiteDatabase database = dbHelper.getWritableDatabase();
dbHelper.resetDatabase(database);

. . .
}
```

Para restablecer la base de datos se puede ver con este código, cómo se instancia la clase "AppStorageHerper", y una vez instanciada, se solicita que se lance el método "resetDatabase()" que internamente borra la información de la base de datos, y la rellena con la información original.

- **Ejercicio de Login**

El desafío de Login, tiene dos pantallas principales, en una se representa el control de acceso, y en la otra un área restringida.

En la pantalla de acceso, la funcionalidad principal que se debe analizar es la lógica que hay detrás de la pulsación del botón “enviar”.

```
protected void logicaBtnLoginSend() {
. . .

if(isNotEmptyBox(userNameEv)){
      if(isNotEmptyBox(userPassEv)){

. . .
}
```

El primer paso a realizar antes de hacer nada es comprobar si los campos de texto están cumplimentados. Para realizar esta comprobación se utiliza la función “isNotEmptyBox()”.

Verificados los campos de texto como rellenos, lo que se hace es solicitar a la base de datos, la comprobación de la existencia del usuario con el identificador y contraseña introducido.

```
protected void logicaBtnLoginSend() {
. . .

String sqlResp =
SQLinjectStorageManager.get_UserLine(getApplicationContext(),
userNameEv.getText().toString(), userPassEv.getText().toString());


. . .
}
```

En este fragmento de código se puede encontrar el primer error de programación introducido de manera deliberada. El texto introducido en el campo de texto, es enviado sin hacer un control previo de lo que almacena. Para solucionar esto debería hacerse una comprobación del texto y verificar que no contiene caracteres problemáticos como la comilla simple: ‘ .

Una vez hecha la comprobación del usuario, la operación que sigue, es la valoración del resultado devuelto en la consulta. Los resultado esperados son “Login fallido”, “Excepción SQLite detectada en la consulta”, “acceso autorizado”

```
protected void logicaBtnLoginSend() {
. . .

if(sqlResp.compareToIgnoreCase("Fail Login")==0){
      helpCounter1++;
      . . .
      if (helpCounter1>3){
            helpCounter1=0;
            showHelp(1);
      }
      . . .

. . .
}
```

Como se ve en el código, si la respuesta a la consulta a la base de datos coincide con “Login fallido”, se aumenta el contador de ayuda número uno, y cuando se alcanzan más de tres fallos de este tipo se muestra la ayuda correspondiente.

```
protected void logicaBtnLoginSend() {
. . .

if(sqlResp.compareToIgnoreCase("Fail Login")==0){
      . . .
}else if(sqlResp.compareToIgnoreCase("A SQLiteException has been
caused, probably that's a good way")==0){
      helpCounter2++;

      if (helpCounter2>4){
            helpCounter2=0;
            showHelp(2);
      }

}else{
      . . .

. . .
}
```

Si la respuesta a la consulta coincide con “Excepción de SQLite”, se aumenta el contador de ayuda número dos, y si alcanzan más de cuatro fallos como este, se muestra la ayuda correspondiente.

```
protected void logicaBtnLoginSend() {
. . .

if(sqlResp.compareToIgnoreCase("Fail Login")==0){
      . . .
}else if(sqlResp.compareToIgnoreCase("A SQLiteException has been
caused, probably that's a good way")==0){
      . . .
}else{
  Intent newIntent = new Intent (getApplicationContext(),
SecureAreaActivity.class);
  newIntent.putExtra("username",userNameEv.getText().toString());
  newIntent.putExtra("password", userPassEv.getText().toString());
  newIntent.putExtra("sqlResp", sqlResp);

  String arrayResp[] = sqlResp.split("\n");
      if(arrayResp.length>1){
            helpCounter3++;
      }

      if (helpCounter3>5){
            helpCounter3=0;
            showHelp(3);
      }
      . . .
}

. . .
}
```

Si la respuesta a la consulta no es ninguna de las dos anteriores, es considerada como correcta, y por tanto se instancia la clase que representará la pantalla del área restringida y se le pasan los valores de la consulta, así como el resultado de esta.

Dado que la consulta es deliberadamente vulnerable, existe la posibilidad de que devuelva más de una línea. Si se retorna más de una línea en la verificación de usuario y contraseña, esto indicaría que se ha sufrido un ataque por inyección de código.

Como se refleja en el código, si se detectan saltos de línea en la respuesta, ello indicará que se ha producido un login incorrecto, y aumentará el contador de errores para este tipo de fallos. A causa de la mala programación deliberada, el proceso de autenticación tendría éxito, y cuando se detectasen más de 5 accesos al área restringida por este medio, se mostraría la ayuda necesaria para que el "alumno" consiguiera entrar como administrador explotando esta vulnerabilidad, y logrando superar el desafío.

La pantalla de acceso restringido, tiene por finalidad dar la bienvenida al usuario a un área que debería ser seguro. A parte de esa bienvenida, se aprovecha para mostrar información de la consulta para que el "alumno" pueda obtener feedback de sus intentos. El detalle principal de esta ventana, que merece explicación, es la forma en la que se detecta la consecución del objetivo.

```
protected void onCreate() {
. . .

String arrayResp[] = sqlResp.split("\n");

tvQuery.setText(R.string.SecureAreaActivity_tv_queryInfo);

if(arrayResp.length==1){
      String arrayResp2[] = arrayResp[0].split("\\|");
      auxUsername = arrayResp2[1];

      if(auxUsername.compareTo("admin")==0){
            tvQuery.setText("");
            sqlResp="";
            ivHacked.setVisibility(ivHacked.VISIBLE);
      }
}
. . .
}
```

Con este fragmento de código, se comprende que para verificar la superación del objetivo, se obtiene el segundo campo de la respuesta que da la base de datos, y se verifica que coincida con el usuario "admin". Comprobada esta coincidencia, se muestra en la pantalla una imagen que informa al "alumno" de que el sistema ha sido asaltado, y por tanto el desafío ha sido superado.

- **Ejercicio de búsqueda de producto**

La función principal que se va a detallar del ejercicio de búsqueda de producto, es el método que contiene la lógica del botón buscar.

Tras obtener la información del campo de búsqueda, lo primero que realiza la función, es lanzar la consulta a la base de datos. Aquí se encuentra una nueva vulnerad introducida, como en el caso anterior, el parámetro se pasa sin ningún tipo de control previo que asegure la ausencia de caracteres especiales como la comilla simple: ' .

```
protected void logicaBtnSearch() {
. . .

String result =
SQLinjectStorageManager.get_Product(getApplicationContext(),
referenceEt.getText().toString());

. . .
}
```

Como se observa en este fragmento de código, la información contenida en el campo de búsqueda se pasa directamente a la consulta.

El siguiente paso a realizar es la verificación del resultado de la consulta, que puede ser del tipo: "producto no encontrado", "Excepción SQLite", o un resultado satisfactorio.

```
protected void logicaBtnSearch() {
. . .

if(result.compareTo("Product not found")==0){
      searchHeaderResultTv.setText(result);
      searchResultTv.setText("");

}else if(result.compareTo("A SQLiteException has been caused,
probably that's a good way")==0){
      helpCounter1++;
      if (helpCounter1>3){
            helpCounter1=0;
            showHelp(4);
      }

      searchHeaderResultTv.setText("");
      searchResultTv.setText(result);
}else{

. . .
}
```

En este fragmento de código, se puede ver, que en primer lugar se comprueba si el resultado devuelto coincide con "producto no encontrado", transmitiéndose ese mensaje al interfaz de resultados.

La segunda comprobación que se hace es más interesante, pues detecta los fallos asociados a excepciones en SQLite. Si se producen más de tres fallos de este tipo, se muestra un consejo para que el "alumno" siga avanzando en su ejercicio.

Si el resultado no coincide con ninguno de estas condiciones, representa que la búsqueda ha sido satisfactoria, y se han obtenido resultados.

```
protected void logicaBtnSearch() {
. . .

if(result.compareTo("Product not found")==0){
      . . .
}else if(result.compareTo("A SQLiteException has been caused,
probably that's a good way")==0){
      . . .
}else{
      if(result.contains(SQLinjectStorageManager.get_AdminPassword(g
etApplicationContext()))){
            ivHacked.setVisibility(ivHacked.VISIBLE);
            searchCongratsTv.setText(R.string.SearchProductActivity_
      tv_congratulations);

      }else{
            helpCounter2++;
            if (helpCounter2>3){
                  helpCounter2=0;
                  helpCounter3++;
                  if(helpCounter3<2){
                        showHelp(5);
                  }else{
                        helpCounter3=0;
                        showHelp(6);
                  }
            }
      }
      . . .

. . .
}


. . .
}
```

En esta ocasión se trata de verificar si se ha logrado superar el desafío. Según el fragmento de código que se muestra, esta verificación se hace comprobando si la contraseña del usuario administrador aparece en el resultado de la búsqueda. En caso de no encontrarse, un par de contadores van aumentando hasta que llegado el momento, se muestran los consejos pertinentes al alumno, de manera que pueda avanzar en el ejercicio.

- **Ejercicio de añadir de producto**

El ejercicio de insertar un nuevo ítem en la tabla de productos, está formado por dos ventanas la primera es la de carga del producto, y la segunda se encarga de mostrar el resultado de dicha carga.

Se comenzará en análisis con la ventana de carga. En esta pantalla la funcionalidad más relevante a destacar es la asociada al botón de búsqueda.

Lo primero que se hace en este método, una vez cargados los valores de los campos en memoria, es verificar si están cumplimentados, o por contrario se encuentra alguno vacío.

Tras esta comprobación, se almacena en una variable, el número de productos almacenados en la base de datos, y a continuación se lanza la instrucción de carga del nuevo producto. Como viene siendo habitual, se ha introducido la vulnerabilidad de no comprobar los campos antes de pasárselos a la instrucción de carga.

```
protected void logicaBtnEnlistProductSave() {
. . .

int numRowsPrev =
Integer.valueOf(SQLinjectStorageManager.get_NumProduct(getApplicatio
nContext()));

boolean error =
SQLinjectStorageManager.set_product(getApplicationContext(),
productNameEv.getText().toString(), eferenceEv.getText().toString(),
priceEv.getText().toString());

if(error){
      helpCounter1++;
      if (helpCounter1>3){
            helpCounter1=0;
            showHelp(7);
      }
}else{
      . . .

. . .
}
```

Cómo se puede ver en el fragmento de código, se almacena en una variable si la ejecución de la instrucción ha producido algún error. En caso afirmativo se irán acumulando dichos errores, y al alcanzar más de tres, se mostrará una ayuda al "alumno".

Si no se producen errores, también se establecerá un contador que se empleará para mostrar otro consejo al "alumno", si pese conseguir inserciones en la base de datos, no está consiguiendo superar el desafío.

```
protected void logicaBtnEnlistProductSave() {
. . .

}else{
      helpCounter2++;
      . . .
      Intent newIntent = new Intent (getApplicationContext(),
ShowProductsActivity.class);

      newIntent.putExtra("numRowsPrev", numRowsPrev);

      newIntent.putExtra("helpCounter2", helpCounter2);

      startActivity(newIntent);

      if (helpCounter2==6){
            helpCounter2=0;
      }
}
. . .
}
```

Como se puede ver en este fragmento de código, la operación de mostrar la ayuda no se realiza en este método. Esto se debe a que se le pasan tanto el contador, como el número de productos a la pantalla de muestra de productos, delegando en ella las tareas de detección de superación de desafío, y de muestra de consejo.

En la pantalla de resultado por tanto, la tarea principal a realizar es la de detectar si se ha superado el desafío o no. Para verificar este hecho, se comprueba el numero de productos antes y después de la ejecución de la instrucción, y si ha aumentado en dos o más, quiere decir que el ejercicio se ha completado con éxito.

```
protected void onCreate(Bundle savedInstanceState) {
. . .

if(numRowsPrev+1<numRowsPost){
      . . .

      tvCongrats.setVisibility(tvCongrats.VISIBLE);
      ivHacked.setVisibility(ivHacked.VISIBLE);

}else{
      int helpCounter=intent.getIntExtra("helpCounter2", -1);

      if (helpCounter==3){
            showHelp(7);
      }
      if (helpCounter==6){
            showHelp(8);
      }
}

. . .
}
```

En este fragmento de código puede verse programadas ambas afirmaciones presentadas más arriba.

- **Ejercicios de Chat**

Sin entrar en detalle sobre la lógica de funcionamiento de la aplicaciones de chat que se salen del objetivo de este proyecto, para estos dos ejercicios restantes, se va a presentar en detalle únicamente la manera en la que se detecta la consecución de los ejercicios.

Tal como ocurre en el resto de ejercicios, la vulnerabilidad introducida es la de no revisar el contenido del campo de mensaje antes de mandárselo a la base de datos. Se descuida de esta manera la aparición de caracteres problemáticos como el ya conocido: ‘ .

<u>Detección en ejercicio de Chat 1</u>

Para detectar que se ha superado el desafío, se comprueban todos los mensajes cuando son cargados en la función fillInAdapter(). En esta función se rellena el “adapter” del chat con todos los mensajes almacenados en la base de datos asociados a esa conversación. La comprobación consiste en verificar si el “token” de usuario aparece en alguno de los mensajes que son cargado en el “adapter”.

```
protected void fillInAdapter(ChatArrayAdapter adapter) {
. . .

if(message_content.contains(context.getText(R.string.token))){
      . . .

      // Inflate and set the layout for the dialog
      builder.setView(inflater.inflate(R.layout.dialog_hacked,
null)).setMessage(R.string.hacked_token_congrats).setPositiveButton(
android.R.string.ok, new DialogInterface.OnClickListener();

      . . .

      // Create the AlertDialog
      AlertDialog dialog = builder.create();
      dialog.show();
}

. . .
}
```

Con este fragmento de código se puede verificar que para detectar que el sistema ha sido vulnerado, y que se ha logado obtener el "token" de usuario, se busca el "token" de usuario dentro del contenido de los mensajes.

Detección en ejercicio de Chat 2

La manera de detectar la superación del desafío en el ejercicio de Chat 2, es muy similar al de Chat 1, pues se realiza también en el método fillInAdapter, y en parte se busca una cadena de texto concreta en cada uno de los mensajes.

La cadena de texto buscada en los mensajes es "You have been hacked", sin embargo para superar el ejercicio no solo debe encontrarse esta cadena, sino que debe encontrarse en el lado del interlocutor. Es decir que debe parecer que dicho mensaje ha sido enviado por el interlocutor, y no por el "alumno".

```
protected void fillInAdapter(ChatArrayAdapter adapter) {
. . .

if((message_content.contains(context.getText(R.string.Chat2Activity_
hack_message_content_detected))==true) &&
(id_user.compareTo(context.getString(R.string.alice_phone)+"&1")==0)
){
      . . .

      // Inflate and set the layout for the dialog
      builder.setView(inflater.inflate(R.layout.dialog_hacked,
null)).setMessage(R.string.hacked_conversation_congrats).setPositive
Button(android.R.string.ok, new DialogInterface.OnClickListener();

      . . .

      // Create the AlertDialog
      AlertDialog dialog = builder.create();
      dialog.show();
}

. . .
}
```

Como puede verse en esta porción de código, para verificar la segunda condición lo que se hace es comprobar si el campo identificativo del usuario con el que se está hablando, termina con el sufijo &1, que indica que se trata de un mensaje proveniente de tal usuario.

- **Interacción con la base de datos**

Para comprender el por qué de las vulnerabilidades introducidas, se detallan a continuación parte de los métodos de interacción con la base de datos, que deliberadamente fueron programados con errores de seguridad.

get UserLine

Con esta función lo que se hace es recuperar de la tabla usuarios el "_id", "user_name", y "description" correspondiente a usuario y contraseña introducidos.

El primer fallo de programación que se comete es confiar en los parámetros que se reciben en la función. Estos parámetros deberían ser analizados para buscar caracteres problemáticos como la comilla simple, o en su defecto enjaular el contenido de los parámetros, y que aunque tuvieran dichos caracteres, no se comprometiera la seguridad de la consulta.

```
String sqlQuery = "SELECT _id, user_name, description FROM users
WHERE user_name='"+userName+"' AND password='"+userPass+"'";

Cursor c = db.rawQuery(sqlQuery, null);
```

Las contramedidas para este tipo de errores son sencillas, basta con filtrar el contenido de los parámetros, o utilizar el método rawQuery() tal como se propone en la web de desarrolladores Android (rawQuery in Android Developer Web, 2014). En dicha explicación, se aclara como se debe usar la función rawQuery, pasándoles una cadena de texto con la consulta a realizar, y un array de cadenas de texto, con los parámetros de la consulta.

get Product

Usando esta función, el objetivo es recuperar de la tabla de productos, los campos: "reference"," product_name", y "value", correspondientes al producto cuya referencia contenga el parámetro indicado.

```
String sqlQuery = "SELECT reference, product_name, value FROM
products WHERE reference LIKE '%"+reference+"%'";

Cursor c = db.rawQuery(sqlQuery, null);
```

Como puede verse por este fragmento de código, el error de programación es el mismo que para el método get_UserLine(), y por tanto las vulnerabilidades y soluciones las mismas también.

set product

En el método set_product(), lo que se trata de realizar es la incorporación de un nuevo producto a la tabla "products".

El fallo de programación que se comete, vuelve a ser el confiar en los parámetros que se reciben en la función. Estos parámetros, como ya se sabe, deberían ser analizados para buscar caracteres problemáticos como la comilla simple, o en su defecto enjaular el contenido de los parámetros, y que aunque tuvieran dichos caracteres, no se comprometiera la seguridad de la consulta.

```
String sql="INSERT INTO products(product_name,reference,value)"+
            " VALUES ('"+productName+"','"+reference+"',"+price+")";

database.execSQL(sql);
```

Para tratar de prevenir este problema, bastaría con filtrar el contenido de los parámetros, o utilizar el método insert() tal como aconseja el autor de este trabajo. Para un correcto uso de esta función se recomienda visitar (SQLite Insert in Android Developer Web, 2014).

set message

En este método, el objetivo es la incorporación de un nuevo mensaje a la tabla "messages" o "messages2", en función si se la llama desde el ejercicio de Chat 1 o el ejercicio de Chat 2.

```
if(chatNumber==1){
      sql="INSERT INTO messages (message_content,id_user,date) "
            + "VALUES ('"+msgContent+"','"+phone+"','"+date+"')";

}else if(chatNumber==2){
      sql="INSERT INTO messages2 (message_content,id_user,date) "
            + "VALUES ('"+msgContent+"','"+phone+"','"+date+"')";
}

database.execSQL(sql);
```

El fallo de programación cometido, por lo que se aprecia en este fragmento de código, es el mismo que para el método set_product(), y por tanto las vulnerabilidades y soluciones las mismas también.

5.1.2. Implementación en Script de búsqueda

En la implementación del Script de análisis de posibles vulnerabilidades, en el código de las aplicaciones, se ha considerado detallar la programación de las funcionalidades a continuación.

- **Listado de aplicaciones**

  La principal tarea de esta función, consiste en crear el listado de aplicaciones Android contenidas en el directorio indicado.

```
fun_list(){
   DATE=`date +%Y-%m-%d_%H:%M`
   FILE=injectSearch_report_$DATE.txt

   echo "**[ InjectSearch Report ]**"
   echo "**[ InjectSearch Report ]**" > $FILE

   ##--> Make the list of apks
   ls $DIR > apks.list
   NUM_APKS=`ls $DIR | wc -l`
   echo $NUM_APKS "apks availables on $DIR"
   echo $NUM_APKS "apks analized on $DIR" >> $FILE
}
```

Se puede ver por el código, que lo primero que se hace es obtener la fecha del sistema con el formato año-mes-día_hora:minutos. El segundo paso realizado es guardar en una variable, el nombre que tomará el informe de resultados. A continuación se muestra por pantalla un mensaje que notifica del inicio del informe, guardando el mismo mensaje en la primera línea del informe en cuestión.

Para obtener la lista de aplicaciones, se hace uso del comando "ls" (Unix ls, 2014), para listar el contenido del directorio que fue pasado como parámetro, y almacenado en la variable "DIR". Este listado se almacena en un archivo de texto llamado apks.list, que será utilizado más adelante para navegar por las aplicaciones Android.

Por último se cuenta el número de aplicaciones y se almacena en la variable "NUM_APKS", la cual es mostrada en un mensaje por pantalla, y almacenada en el informe, para que éste sea lo más completo posible.

- **Decompilado de aplicaciones**

La principal tarea de esta función es la de decompilar todas las aplicaciones Android contenidas en el directorio indicado.

```
fun_decompile(){
   echo "----<[ Decompiling Process ]>----" >> $FILE
   ##--> Read the list of apks and decopile each one
   for line in $(cat apks.list);
   do

      # if $DDIR/$line is a directory ...
      if test -d $DDIR/$line
      then
         echo "$line was decompiled before"
         echo "$line was decompiled before" >> $FILE
      else
         printf "\n\n --> DECOMPILING: [$line]"
         apps/apktool -q d $DIR/$line $DDIR/$line
         echo "$line has been decompiled" >> $FILE
      fi
   done
}
```

Analizando el código de esta función, se puede ver que el primer paso consiste en introducir una línea en el informe de resultados, que referencie el inicio del proceso de decompilación. El siguiente paso a seguir es recorrer el listado de aplicaciones creado en la función de listado. Cada línea de la lista corresponde a una aplicación Android, y para cada una de ellas, se realizan los siguientes pasos:

- Comprobar que la aplicación ha sido decompilada previamente.
- Si la aplicación ya fue decompilada se notifica por pantalla y se añade una nueva línea al informe registrando el mismo hecho.
- Si la aplicación no fue decompilada con anterioridad, se notifica por pantalla del inicio del proceso de decompilación, y se utiliza la herramienta Apktool para que se realice tal operación. Concluida la decompilación, se añade una nota informativa en el informe de resultados.

- **Búsqueda de posibles vulnerabilidades**

La principal tarea de esta función, consiste en la de búsqueda de posibles vulnerabilidades en el código fuente de las aplicaciones Android, decompiladas en el directorio indicado.

La función de búsqueda de posibles vulnerabilidades es la más compleja del Script. La idea de esta tarea es recorrer todas las aplicaciones decompiladas, y buscar en su código fuente, cadenas de texto que demuestren, un uso inadecuado de los métodos proporcionados por el API de Android, para realizar consultas a la base de datos SQLite.

El uso inadecuado de los métodos en cuestión se debe, a que en vez de utilizar los métodos de enjaulado que proporciona Android, se realizan las consultas a mano, pasándole las variables de manera directa. Estos ejemplos de mal uso, son los que posibilitan los ataques por SQL Injection, y los que se tratarán de buscar en las aplicaciones para analizar.

Los ejemplos de mal uso que se van a buscar en el código fuente de las aplicaciones serán las siguientes expresiones[12]:

- Para las instrucciones de INSERT:
  "INSERT " | " INTO " | " VALUES" | "\\\'" | NOT ” \'[[:print:]]+\'”

- Para las instrucciones de SELECT:
  "SELECT " | " FROM " | "\\\'" | NOT "\'[[:print:]]+\'"

- Para las instrucciones de UPDATE:
  "UPDATE " | " SET " | "\\\'" | NOT "\'[[:print:]]+\'"

- Para las instrucciones de DELETE:
  "DELETE " | " FROM " | "\\\'" | NOT "\'[[:print:]]+\'"

[12] Para más información sobre el uso de expresiones regulares en el comando grep, consultar: (Regular expressions in grep, 2010)

```
fun_search(){
    echo "----<[ Scanning Process ]>----" >> $FILE
    ##--> Read the list of apks and search vulnerabilities
    for line in $(cat apks.list);
    do
        printf "\n\n --> SCANNING: [$DDIR/$line]"
        printf "\n\n --> SCANNING: [$DDIR/$line]\n" >> $FILE

        printf "\n[INSERT Clauses]\nPosible Vulnerabilities:" >> $FILE
        grep -Ri "insert " $DDIR/$line | grep -i " into " | grep -i " values" | grep -i "\\\'" | grep -Ev
\'[[:print:]]+\' | wc -l >> $FILE
        grep -Ri "insert " $DDIR/$line | grep -i " into " | grep -i " values" | grep -i "\\\'" | grep -Ev
\'[[:print:]]+\' >> $FILE


        printf "\n[SELECT Clauses]\nPosible Vulnerabilities:" >> $FILE
        grep -Ri "select " $DDIR/$line | grep -i " from " | grep -i "\\\'" | grep -Ev \'[[:print:]]+\' | wc -
l >> $FILE
        grep -Ri "select " $DDIR/$line | grep -i " from " | grep -i "\\\'" | grep -Ev \'[[:print:]]+\' >>
$FILE


        printf "\n[UPDATE Clauses]\nPosible Vulnerabilities:" >> $FILE
        grep -Ri "update " $DDIR/$line | grep -i " set " | grep -i "\\\'" | grep -Ev \'[[:print:]]+\' | wc -l
>> $FILE
        grep -Ri "update " $DDIR/$line | grep -i " set " | grep -i "\\\'" | grep -Ev \'[[:print:]]+\' >>
$FILE


        printf "\n[DELETE Clauses]\nPosible Vulnerabilities:" >> $FILE
        grep -Ri "delete " $DDIR/$line | grep -i " from " | grep -i "\\\'" | grep -Ev \'[[:print:]]+\' | wc
-l >> $FILE
        grep -Ri "delete " $DDIR/$line | grep -i " from " | grep -i "\\\'" | grep -Ev \'[[:print:]]+\' >>
$FILE
    done
    printf "\n"

    printf "\nEND OF SEARCH\n\n"
}
```

Estudiando el código de esta función, podemos ver que el primer paso que se realiza es introducir una nueva línea en el informe de resultados, reflejando el inicio de la fase de búsqueda de vulnerabilidades. Hecho esto, se pasa a recorrer el listado de aplicaciones, y para cada una de las aplicaciones enumeradas, se realizan los siguientes pasos:

- Se notifica por pantalla que se está analizando el programa [nombre_de_programa], y se refleja en mismo hecho en el informe de resultados.
- Se añade una nueva línea al informe estableciendo que a partir de este momento se reflejarán vulnerabilidades en operaciones INSERT.
- Con la ayuda del comando grep, y el comando wc (Unix wc, 2014), se buscan coincidencias con la expresión presentada arriba, se cuenta el número de estas, y se añade todo ello en el informe de resultados.
- Para las instrucciones SELECT, UPDATE, y DELETE, se realizan estos dos últimos pasos, con las características especificas de dichas instrucciones.

Recorridas todas las aplicaciones, añade la última línea al informe, en la que se refleja que el proceso de búsqueda ha concluido.

5.2. Resultado del plan de pruebas.

El resultado obtenidos al realizar la batería de pruebas establecidas en el apartado 2.8 ha sido satisfactorio en todas y cada una de las pruebas, tal y como se refleja en la tabla a continuación.

Identificador de Prueba	Resultado
PR-A-01	Superada
PR-A-02	Superada
PR-A-03	Superada
PR-A-04	Superada
PR-A-05	Superada
PR-S-01	Superada
PR-S-02	Superada
PR-S-03	Superada
PR-S-04	Superada
PR-S-05	Superada
PR-S-06	Superada

Tabla 71. Resultado del plan de pruebas.

Capítulo 6. Gestión del proyecto

Este capítulo presentará la planificación de esfuerzo inicial y final del proyecto mediante el uso de diagramas de Gantt. A continuación se detallarán los medios técnicos empleados en el desarrollo del proyecto, ya sea en hardware o software. Para terminar el capítulo, se realizará un análisis económico del proyecto, donde se presentará una estimación de costes, y por último se analizará la forma con la que obtener ingresos por la aplicación.

6.1. Planificación Inicial y esfuerzo real

En este apartado, se enumerarán las tareas que han compuesto el proyecto, detallando el esfuerzo dedicado en cada una ellas. Para representar estos esfuerzos, se adjuntarán la planificación de esfuerzo inicial, y el esfuerzo final, mediante la representación de diagramas de Gantt. Teniendo estas dos planificaciones se hará una valoración de si la planificación inicial del proyecto se cumplió o sufrió algún tipo de desviación.

6.1.1. Planificación inicial

A continuación se enumeran las distintas tareas que inicialmente fueron planificadas para ser desarrolladas en el proyecto. Tal como podrá observarse, esta planificación encontrará ciertas diferencias con la de esfuerzo real cumplido.

- **Inicio**
  - **Lista de objetivos:** Se enumeran los objetivos buscados por el proyecto, que deberán ser cumplidos a la finalización de este.
- **Análisis**
  - **Estado de la cuestión:** Se realiza una valoración de la situación actual en lo referente a la temática del proyecto. Se analizan las alternativas de almacenamiento en los dispositivos móviles, y las herramientas de concienciación disponibles.
  - **Valoración de alternativas de diseño:** Se realiza un estudio de las posibles plataformas sobre las que desarrollar.
  - **Definición de casos de uso:** Se establecen los casos de uso que cubren el proyecto.
  - **Definición de requisitos:** Se definen los requisitos a cumplir por el proyecto.
  - **Definición de pruebas:** Se establece el plan de pruebas que verifica la cobertura de todos y cada uno de los requisitos definidos.
- **Diseño**
  - **Diseño de arquitectura:** Se estudia la arquitectura que utilizar en el desarrollo del proyecto.
  - **Diseño de componentes:** Se definen los componentes que deben integrar el proyecto.
  - **Diseño de funciones principales:** Se definen las principales funciones que debe implementar el proyecto.

- **Implementación**
  - **Fase de programación:** Se realiza la programación de los elementos del proyecto.
- **Pruebas**
  - **Ejecución de pruebas:** Se ejecutan las pruebas definidas en el plan que las recoge.
- **Estudio:** Realización del estudio de vulnerabilidades SQLite en aplicaciones Android.
- **Documentación:** Realización de este documento.

A continuación en la Figura 19, se puede ver el diagrama de Gantt realizado con la primera planificación.

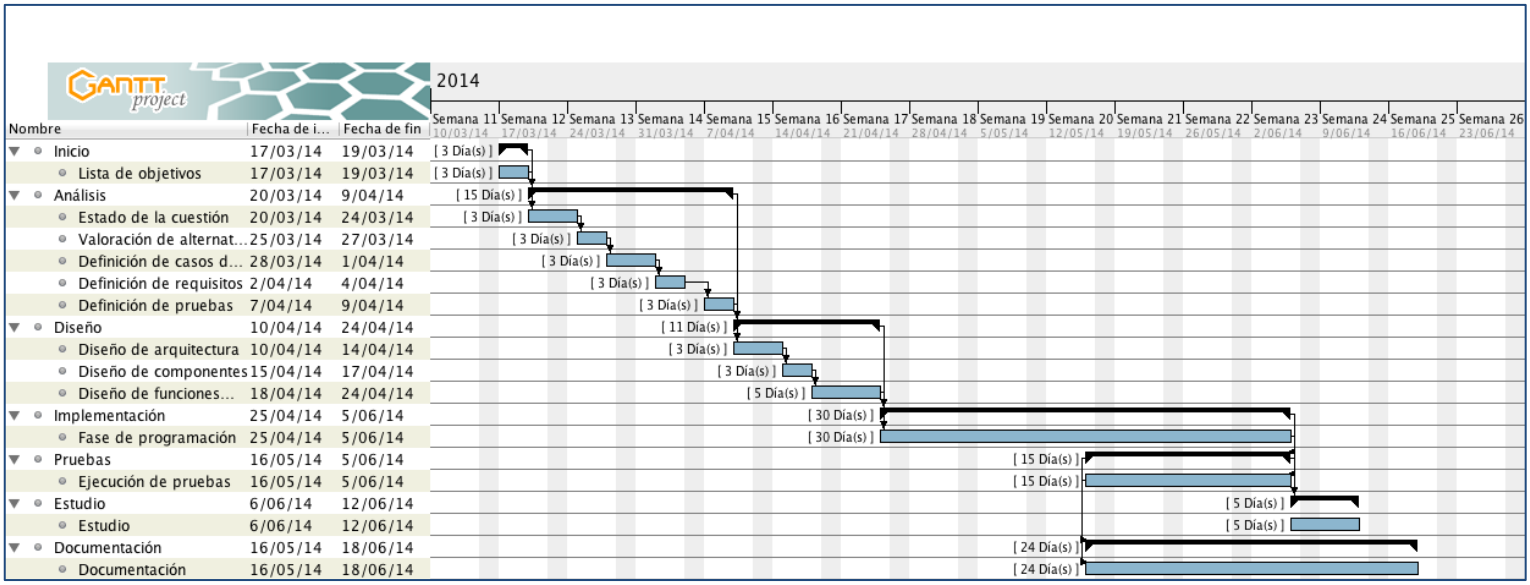

Figura 19. Diagrama de Gantt con Planificación Inicial

6.1.2. Planificación Real

La planificación inicial planteada, no se cumplió tal como estaba previsto, pues no fue necesario emplear tanto esfuerzo en las tareas de lista de objetivos, análisis, y diseño. En la fase de implementación en cambio, debido a la carga sufrida por proyectos externos a este, se produjo un esfuerzo extra de diez días. Este esfuerzo adicional también se vio reflejado en la fase de pruebas, que aumentó en cinco días respecto a su planificación inicial, y en la fase de documentación donde aumentó en otros 6 días más. Por otra parte, en la fase de estudio se logró recortar un día de esfuerzo respecto a lo inicialmente previsto.

A continuación en la Figura 20, se puede ver el diagrama de Gantt utilizando los resultados reales de esfuerzo realizado.

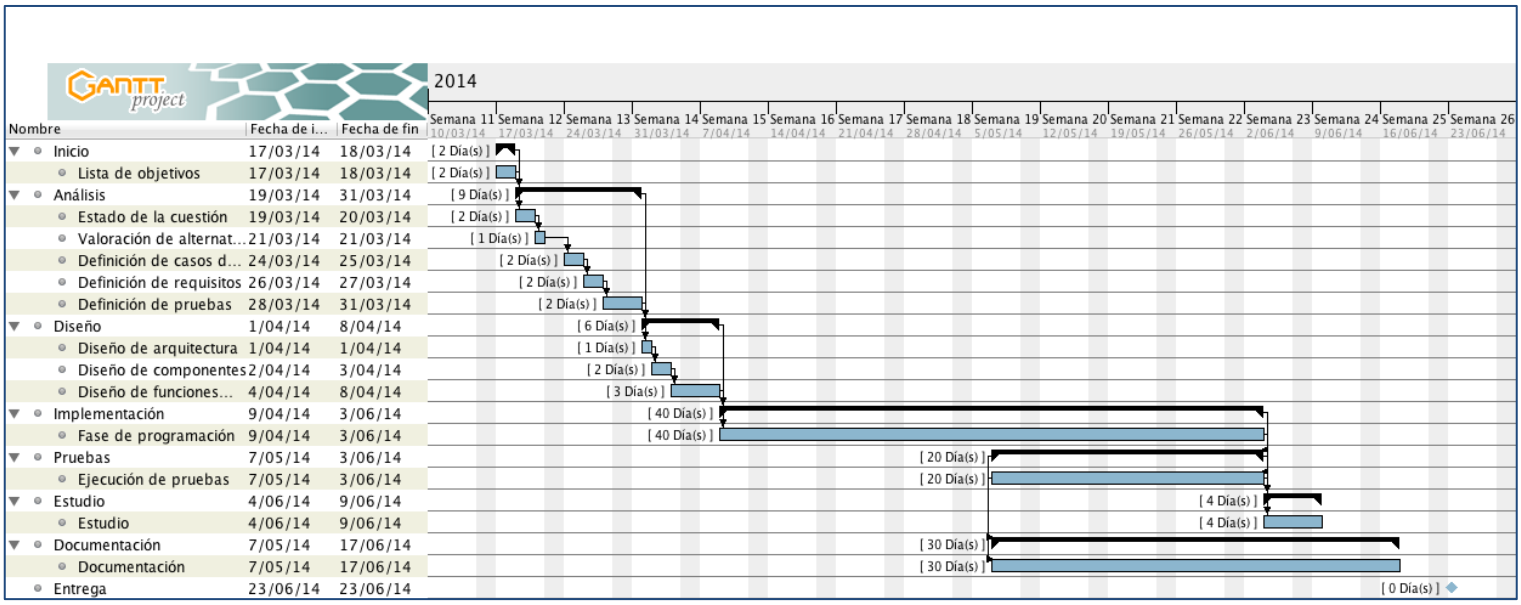

Figura 20. Diagrama de Gantt con el Esfuerzo Final Realizado.

6.2. Medios técnicos

En este apartado se detallan los medios técnicos utilizados, tanto hardware como software, en la elaboración del proyecto.

6.2.1. Hardware

Para la elaboración de este proyecto se han utilizado los dispositivos hardware reflejados en la siguiente tabla.

Tipo	Nombre	Descripción
Ordenador portátil	Apple MacBook	Procesador Intel Core 2 Duo 2,4GHz, 4GB de RAM, Mac OS X 10.7.5
Teléfono móvil	Samsung Galaxy S	Procesador ARM Cortex-A8 1GHz, 512 MB de RAM, Android 4.3.1
Teléfono móvil	BQ Aquaris 5	Procesador ARM Quad Core Cortex A7 1,2GHz, 1GB de RAM, Android 4.2.1
Monitor	BenQ GL2250HM	LED, 21,5 pulgadas
Teclado y ratón	Logitech Wireless Combo MK520	-

Tabla 72. Hardware Utilizado.

6.2.2. Software

En la tabla a continuación se refleja el software utilizado para la realización de este proyecto.

Tipo	Nombre	Página Web
Sistema operativo	Apple Mac OS X 10.7.5	https://www.apple.com/es/osx/
Entorno de Programación	Android Developer Tools	https://developer.android.com/tools/index.html
Editor de textos	Sublime Text 2	http://www.sublimetext.com/
Procesador de textos	Microsoft Office Word 2011	http://www.microsoft.com/spain/mac
Editor de presentaciones	Microsoft Office Power Point 2011	http://www.microsoft.com/spain/mac
Diagramas	yED	http://www.yworks.com/en/products_yed_about.html
Diagramas	Visual Paradigm Modeler	http://www.visual-paradigm.com/
Diagramas	GanttProject 2.6.1	http://www.ganttproject.biz/
Copias de seguridad / Respaldo	Dropbox 2.8.2	https://www.dropbox.com/
Decompilador	Apktool	https://code.google.com/p/android-apktool/

Tabla 73. Software Utilizado.

6.3. Análisis económico

En este apartado se detallará cada uno de los costes que suponen la realización del proyecto. Los costes derivados de este proyecto se componen de los asociados a RRHH, Hardware, y Software.

Para conocer a desviación de costes sufridas, se comparará el presupuesto inicial y los gastos finales reales del proyecto.

Para concluir la sección se realizará el estudio de la estrategia para venta, más acorde al tipo de proyecto desarrollado.

6.3.1. Metodología de estimación de costes

Para realizar el cálculo de los costes estimados asociados al proyecto, se han agrupado cuatro categorías: personal, hardware, software y costes indirectos. La finalidad de usar estos costes, no es otra que ayudar a estimar, de una forma relativamente fiable, los gastos que supondrá el proyecto.

- **Coste de personal**
  Los costes personales englobarán los gastos derivados de pagar el salario del ingeniero informático a cargo del proyecto. El salario establecido se basará en los perfiles definidos por la Asociación de Titulados Universitarios Oficiales en Informática (ALI perfiles, 2011). Se debe señalar que dicha asociación no aporta tabla de salarios recomendados para los distintos perfiles profesionales, por lo que se establece un salario medio de ingeniero junior.
- **Coste de hardware**
  Para el cálculo del hardware, se emplearán los costes de las facturas de los elementos correspondientes.
- **Coste de software**
  En el caso de los costes por software, se emplearan los precios de las licencias necesarias para el desarrollo del proyecto.
- **Costes indirectos**
  Los costes indirectos estarán formados por aquellos costes que no aportan nada directamente al proyecto, pero que son necesarios para los elementos de desarrollo de éste. En este caso se valorarán los gastos de luz y de internet.

6.3.2. Análisis de costes estimados

En este apartado se mostrarán los cálculos estimados antes de producirse el desarrollo del proyecto.

Estimación de coste personal

Los costes de personal o RRHH se componen únicamente de los honorarios del ingeniero informático encargado del desarrollo del proyecto. La planificación inicial del proyecto, valoró una carga diaria de 6 horas durante 68 días.

Concepto	Horas	Honorarios	Coste RRHH
Ingeniero Informático	408 horas	20 €/hora	8160 €

Tabla 74. Coste de Personal Estimado.

Estimación de coste hardware

A continuación se especificará el coste en Hardware destinado al desarrollo del proyecto. Todos los precios tienen incluido el 21% de IVA.

Concepto	Unidades	Precio unitario	Vida útil estimada	Tiempo de uso	Coste para el proyecto
Apple MacBook	1	614,5 €	60 meses	2,27 meses	23,25 €
Samsung Galaxy S	1	189 €	36 meses	2,27 meses	11,92 €
BQ Aquaris 5	1	199 €	36 meses	2,27 meses	12,55 €
BenQ GL2250HM	1	105,2 €	60 meses	2,27 meses	3,98 €

Concepto	Unidades	Precio unitario	Vida útil estimada	Tiempo de uso	Coste para el proyecto
Logitech Wireless Combo MK520	1	44,7 €	60 meses	2,27 meses	1,70 €
TOTAL					**53,4 €**

Tabla 75. Coste de Hardware Estimado.

Estimación de coste software

A continuación se especificará el coste en Software destinado al desarrollo del proyecto. Todos los precios tienen incluido el 21% de IVA.

Concepto	Unidades	Precio Licencia	Vida útil estimada	Tiempo de uso	Coste para el proyecto
Apple Mac OS X 10.7.5	1	25 €	12 meses	2,27 meses	4,73 €
Android Developer Tools	1	0 €	-	2,27 meses	0 €
Sublime Text 2	1	0 €	-	2,27 meses	0 €
Office 2011	1	119 €	36 meses	2,27 meses	7,51 €
yED	1	0 €	-	2,27 meses	0 €
Visual Paradigm Modeler	1	73,12 €	48 meses	2,27 meses	3,46 €
GanttProject 2.6.1	1	0 €	-	2,27 meses	0 €
Dropbox 2.8.2	1	0 €	-	2,27 meses	0 €
Apktool	1	0 €	-	2,27 meses	0 €
TOTAL					**15,7 €**

Tabla 76. Coste de Software Estimado.

Estimación de costes indirectos

Para el cálculo de costes indirectos, se han valorado los costes de luz e internet. Para el gasto de internet se ha valorado una de las opciones disponibles en el mercado cuyo importe asciende a unos 47€ mensuales. Respecto del consumo de la luz se ha hecho un cálculo mensual de unos 65€.

Concepto	Precio mensual	Tiempo de uso	Coste para el proyecto
Conexión a internet	47 €	2,27 meses	106,70 €
Luz	65 €	2,27 meses	147,55 €
TOTAL			**254,25 €**

Tabla 77. Costes Indirectos Estimados.

Estimación del costes total
El cálculo de costes acumulados en la estimación inicial del proyecto ha sido el representado en la tabla a continuación.

Concepto	Importe
Coste de personal	8160 €
Coste de hardware	53,4 €
Coste de software	15,7 €
Costes indirectos	254,25 €
TOTAL	**8483,35 €**

Tabla 78. Coste Total Estimado.

6.3.3. Análisis de costes reales
En este aparado se muestran los cálculos resultantes de los costes reales del proyecto una vez este ha concluido.

Coste personal real
Los costes de personal o RRHH se componen únicamente de los honorarios del ingeniero informático encargado del desarrollo del proyecto. La planificación real del proyecto, computó una carga media diaria de 5,75 horas durante 67 días.

Concepto	Horas	Honorarios	Coste RRHH estimado	Coste RRHH real	Diferencia
Ingeniero Informático	385,25 horas	20 €/hora	8160 €	7705 €	455 € (-)

Tabla 79. Coste de Personal Real.

Como se puede ver en esta tabla, el número de horas finalmente invertidas han sido 385,25, frente a las 408 previstas en un inicio. Esto refleja un pequeño ahorro en los costes de personal. En el resto de costes también se verá reflejada una pequeña reducción, pero debido a que solo se debe a un día de diferencia, dicho ahorro será despreciable.

Coste hardware real
En cuanto al hardware, software, y costes indirectos, la diferencia se encuentra en los prorrateos de costes, pero al tratarse solo de un día de diferencia apenas hay cambios relevantes.

Concepto	Unidades	Precio unitario	Vida útil estimada	Tiempo de uso	Coste HW estimado	Coste HW real	Diferencia
Apple MacBook	1	614,5 €	60 meses	2,23 meses	23,25 €	22,84 €	0,41 € (-)
Samsung Galaxy S	1	189 €	36 meses	2,23 meses	11,92 €	11,71 €	0,21 € (-)
BQ Aquaris 5	1	199 €	36 meses	2,23 meses	12,55 €	12,37 €	0,18 € (-)
BenQ GL2250HM	1	105,2 €	60 meses	2,23 meses	3,98 €	3,91 €	0,07 € (-)
Logitech Wireless Combo MK520	1	44,7 €	60 meses	2,23 meses	1,70 €	1,67 €	0,03 € (-)
TOTAL					**53,4 €**	**52,5 €**	**0,9 €** (-)

Tabla 80. Coste de Hardware Real.

Coste software real

Concepto	Unidades	Precio Licencia	Vida útil estimada	Tiempo de uso	Coste SW estimado	Coste SW real	Diferencia
Apple Mac OS X 10.7.5	1	25 €	12 meses	2,23 meses	4,73 €	4,65 €	0,08 € (-)
Android Developer Tools	1	0 €	-	2,23 meses	0 €	0 €	0 €
Sublime Text 2	1	0 €	-	2,23 meses	0 €	0 €	0 €
Office 2011	1	119 €	36 meses	2,23 meses	7,51 €	7,32 €	0,19 € (-)
yED	1	0 €	-	2,23 meses	0 €	0 €	0 €
Visual Paradigm Modeler	1	73,12 €	48 meses	2,23 meses	3,46 €	3,40 €	0,06 € (-)
GanttProject 2.6.1	1	0 €	-	2,23 meses	0 €	0 €	0 €
Dropbox 2.8.2	1	0 €	-	2,23 meses	0 €	0 €	0 €
Apktool	1	0 €	-	2,23 meses	0 €	0 €	0 €
TOTAL					**15,7 €**	**15,37 €**	**0,33 €** (-)

Tabla 81. Coste de Software Real.

Costes indirectos reales

Concepto	Precio mensual	Tiempo de uso	Coste estimado	Coste real	Diferencia
Conexión a internet	47 €	2,23 meses	106,70 €	104,81 €	1,89 € (-)
Luz	65 €	2,23 meses	147,55 €	144,95 €	2,60 € (-)
TOTAL			**254,25 €**	**249,76 €**	**4,49 €** (-)

Tabla 82. Costes Indirectos Reales.

Estimación del costes total

El cálculo de costes acumulados en la estimación resultante del proyecto y la diferencia respecto a los costes estimados, han sido los representados en la tabla a continuación.

Concepto	Importe estimado	Importe real	Diferencia
Coste de personal	8160 €	7705 €	455 € (-)
Coste de hardware	53,4 €	52,5 €	0,9 € (-)
Coste de software	15,7 €	15,37 €	0,33 € (-)
Costes indirectos	254,25 €	249,76 €	4,49 € (-)
TOTAL	**8483,35 €**	**8022,63 €**	**460,72 €** (-)

Tabla 83. Costes Totales y Finales de Proyecto.

Como se observa en la tabla, los costes finales del proyecto han resultado 8022,63€, menos que los 8483,35€ previstos inicialmente, y cuya desviación resultante son 460,72€. Esta desviación supone un 5,43% de ahorro sobre el presupuesto inicial. La desviación producida es bastante pequeña, y esto se debe a que pese a recortar en tiempo en muchas actividades, la dedicación de más esfuerzo del previsto en la fase de programación, pruebas y documentación han hecho que el cómputo total de días sea prácticamente el mismo.

6.3.4. Análisis de la forma de venta de la aplicación

La finalidad del desarrollo de este proyecto es principalmente social, y educativo, por lo que la obtención de beneficios del mismo se estudiará mediante un sistema de donaciones.

Las aplicaciones del proyecto, se distribuirán bajo la licencia pública de GNU o GNU General Public License en inglés (GNU GPL, 2007). Este tipo de licencia favorece el uso, distribución y modificación de software libre, y se ha elegido, para que otros desarrolladores puedan colaborar en el futuro, y añadir nuevas funcionalidades a los mismos.

El retorno de la inversión o ROI esperado, basándose en los consejos presentados en (Denney, 2003) son del 30%, por lo que el importe que se espera recaudar para rentabilizar la inversión es de 10430 €[13].

El rango de cantidades prestablecidas para las donaciones que se establece son las siguientes: 1 €, 3 €, 5 € o 10 €. También se aceptarán donaciones de más de 10 € en las que las cantidades ya dejarán de estar prestablecidas.

La tabla a continuación representa la cantidad de donaciones que se esperan recibir, en función de los distintos importes, para conseguir llegar al objetivo del ROI.

Importe	Donaciones necesarias	Donaciones al día	Meses para ROI
1 €	10430	10	35
3 €	3477	10	12
5 €	2086	10	7
10 €	1043	10	4

Tabla 84. Meses para ROI del 30% en importes de donaciones prestablecidos.

Realizando una media aritmética de las donaciones se obtiene los resultados de la siguiente tabla.

Importe	Donaciones necesarias	Donaciones al día	Meses para ROI
4,75 €	2196	10	8

Tabla 85. Mese para ROI del 30% en donaciones para un importe medio.

Si se lograra conseguir la donación media expuesta, de unas 10 donaciones con importe medio de 4,75 € diarios, lo consecución del ROI llegaría a los ocho meses. El importe recaudado al final del octavo mes sería de unos 11400 €, y el beneficio real del proyecto sumaría 3377,37 €

Dado que se trata de un proyecto social y educativo, se buscaría la colaboración con las universidades y centros de estudios tecnológicos avanzados, para que tanto el software desarrollado cómo la web del proyecto, y el sistema de donaciones estuviera alojado en servidores de dichas organizaciones, ahorrando así para el proyecto dichos costes de mantenimiento.

[13] El cálculo del importe resulta de la operación: 8022,63 * 1,30 ≅ 10430 €

Capítulo 7. Estudio

Para complementar el trabajo de las aplicaciones desarrolladas, se ha decido realizar un estudio de los programas que se encuentran disponibles en el mercado de aplicaciones Android, y analizar si cometen los errores de programación estudiados en este proyecto.

7.1. Objetivos del estudio

El objetivo del estudio es conocer mediante el análisis de una muestra de aplicaciones Android, si los desarrolladores de aplicaciones realizan un buen uso de las funciones que proporciona el API de Android, para gestionar la información almacenada en bases de datos SQLite.

7.2. Planteamiento del estudio

En la realización del estudio se utilizará una muestra de aplicaciones disponibles para sistemas Android. Esta muestra incluirá aplicaciones de categorías distintas, desde finanzas o productividad, hasta juegos o aplicaciones deportivas. Adicionalmente a estas aplicaciones legitimas procedentes del market de Android, se estudiará también la programación en una muestra de malware, para comprobar si los delincuentes informáticos también cometen errores de esta clase.

Lo que se trata de detectar en este estudio son los fallos de seguridad expuestos. Estos fallos suelen desembocar en la vulnerabilidades de SQL injection comentadas, pero la aparición de estos fallos no indica obligatoriamente que la vulnerabilidad pueda ser explotable. Para que la vulnerabilidad sea explotable, debemos recordar que tiene que estar asociada directamente a alguna consulta o instrucción lanzada desde un formulario.

Un total de 27 categorías con entre 6 o 7 aplicaciones cada una, nos proporcionan una base de 177 aplicaciones legitimas a analizar. Además un total de 32 aplicaciones malintencionadas se proporcionaron para su análisis. El listado de estas aplicaciones puede ser consultado en el ANEXO 2. Listado de Aplicaciones Analizadas en el Estudio.

7.3. Pruebas realizadas

En este apartado se explicará en qué consiste la prueba que se pasará a las aplicaciones listadas en el apartado anterior. Para demostrar el buen funcionamiento de la metodología, se analizará la aplicación desarrollada por el autor de este proyecto, de la cual son conocidas sus vulnerabilidades.

La prueba que se realizará a las aplicaciones consiste en ejecutar la aplicación de análisis desarrollada por el autor del proyecto, y comprobar en el informe, en qué tipo de instrucciones SQL se comete, en qué clase de la aplicación se encuentra, y que número de ellas hay.

7.3.1. Caso de prueba

El caso de prueba, tal como se ha adelantado, se procede al análisis de la aplicación desarrollado por el usuario: SQLinject.apk

Se utiliza la herramienta de búsqueda de posibles vulnerabilidades y el resultado obtenido es:

```
--> SCANNING: [dapks/SQLinject.apk]

[INSERT Clauses]
Posible Vulnerabilities:      6
dapks/SQLinject.apk/smali/com/ljulian/storage/AppStorageHelper.smali:    const-string
v6, "INSERT INTO personal_data (token,nick,phone) VALUES (\'"
dapks/SQLinject.apk/smali/com/ljulian/storage/AppStorageHelper.smali:    const-string
v6, "INSERT INTO messages (id_user,date,message_content) VALUES (\'1&"
dapks/SQLinject.apk/smali/com/ljulian/storage/AppStorageHelper.smali:    const-string
v6, "INSERT INTO messages2 (id_user,date,message_content) VALUES (\'1&"
dapks/SQLinject.apk/smali/com/ljulian/storage/SQLinjectStorageManager.smali:
const-string v8, "INSERT INTO messages (message_content,id_user,date) VALUES (\'"
dapks/SQLinject.apk/smali/com/ljulian/storage/SQLinjectStorageManager.smali:
const-string v8, "INSERT INTO messages2 (message_content,id_user,date) VALUES (\'"
dapks/SQLinject.apk/smali/com/ljulian/storage/SQLinjectStorageManager.smali:
const-string v8, "INSERT INTO products(product_name,reference,value) VALUES (\'"

[SELECT Clauses]
Posible Vulnerabilities:      2
dapks/SQLinject.apk/smali/com/ljulian/storage/SQLinjectStorageManager.smali:
const-string v8, "SELECT reference, product_name, value FROM products WHERE
reference LIKE \'%"
dapks/SQLinject.apk/smali/com/ljulian/storage/SQLinjectStorageManager.smali:
const-string v8, "SELECT _id, user_name, description FROM users WHERE
user_name=\'"

[UPDATE Clauses]
Posible Vulnerabilities:      0

[DELETE Clauses]
Posible Vulnerabilities:      0
```

Tal como muestra el informe, se detectan seis vulnerabilidades de tipo INSERT, tres en la clase com.ljulian.storage.AppStorageHelper, y otras tres en com.ljulian.storage.SQLinjectStorageManager. Además de estas posibles vulnerabilidades, el script de análisis también encuentra otras dos posibles vulnerabilidades de tipo SELECT en la clase com.ljulian.storage.SQLinjectStorageManager.

Ya que conocemos las vulnerabilidades introducidas (Capítulo 4), basta consultar este capítulo para verificar que dichas vulnerabilidades existen y son explotables.

7.4. Resultados obtenidos

En este apartado mostraremos los resultados obtenidos, tras el análisis de todas las aplicaciones enumeradas. La tabla de resultados generales está disponible para su consulta en ANEXO 3. Tabla de Resultados del Estudio.

Se analizaron un total de 177 aplicaciones obtenidas de la plataforma oficial de descargas de Android. Otras 32 aplicaciones de malware adicionales, participaron análisis. Estas aplicaciones de malware, fuero descargadas de plataformas no oficiales. En total se ha reunido para el análisis un total de 209 aplicaciones Android (Figura 21).

Figura 21. Gráfico de Aplicaciones Analizadas.

De entre estas 209 aplicaciones, se ha obtenido un resultado positivo del 20% de aplicaciones con posibles vulnerabilidades, como se muestra en la Figura 22.

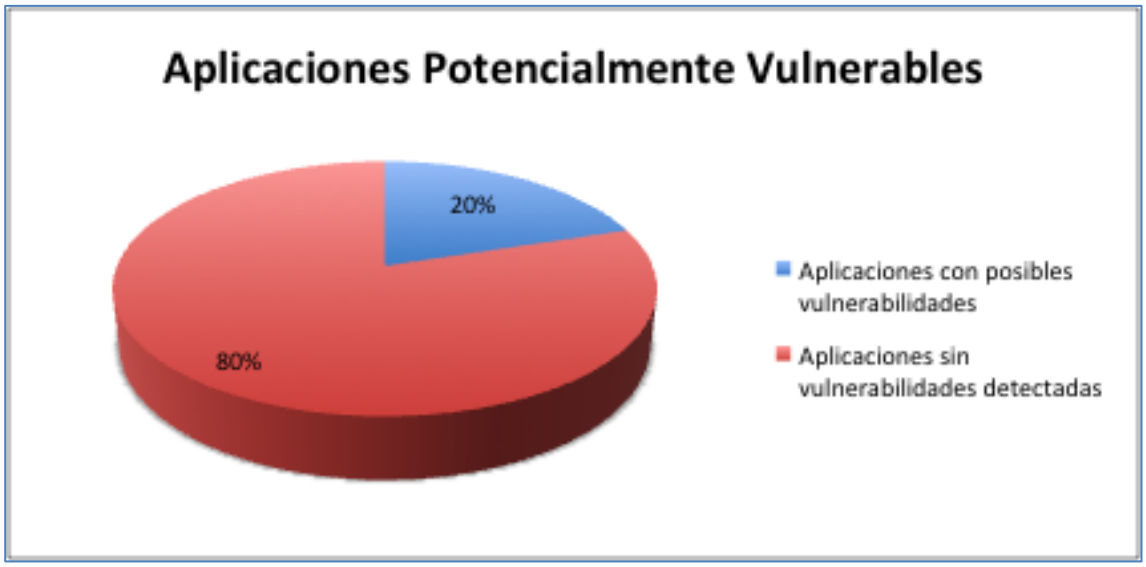

Figura 22. Gráfico de Aplicaciones Potencialmente Vulnerables.

Analizando los resultado, se ha podido observar que se comete un pequeño número de casos de falsos positivos. Eliminando estos falsos positivos detectados, el porcentaje de aplicaciones con potenciales vulnerabilidades es del 16%, el cual se refleja en el gráfico de la Figura 23.

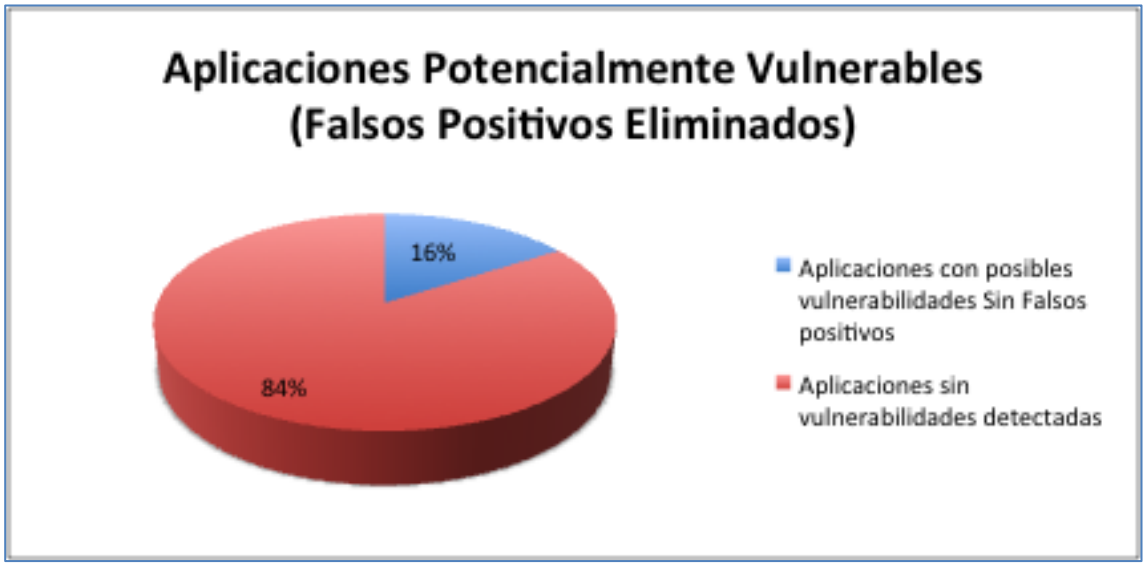

Figura 23. Gráfico de Aplicaciones Potencialmente Vulnerables con Falsos Positivos Eliminados.

Para conocer el grado de veracidad de los resultados, y descubrir incidencia de estos falsos positivos, se ha calculado el número de estos, y ha resultado que solo el 6% de los positivos señalados por la aplicación, no son positivos realmente(Figura 24).

Figura 24. Gráfico de Veracidad de Resultados.

Al estudiar más en profundidad en estos falsos positivos se detectó que gran parte de ellos se producían por la clase de un módulo publicitario de “admod”. Estos

falsos positivos, representan un total del 80% de los falsos positivos cometidos (Figura 25), y podrían ser fácilmente corregidos para futuras versiones.

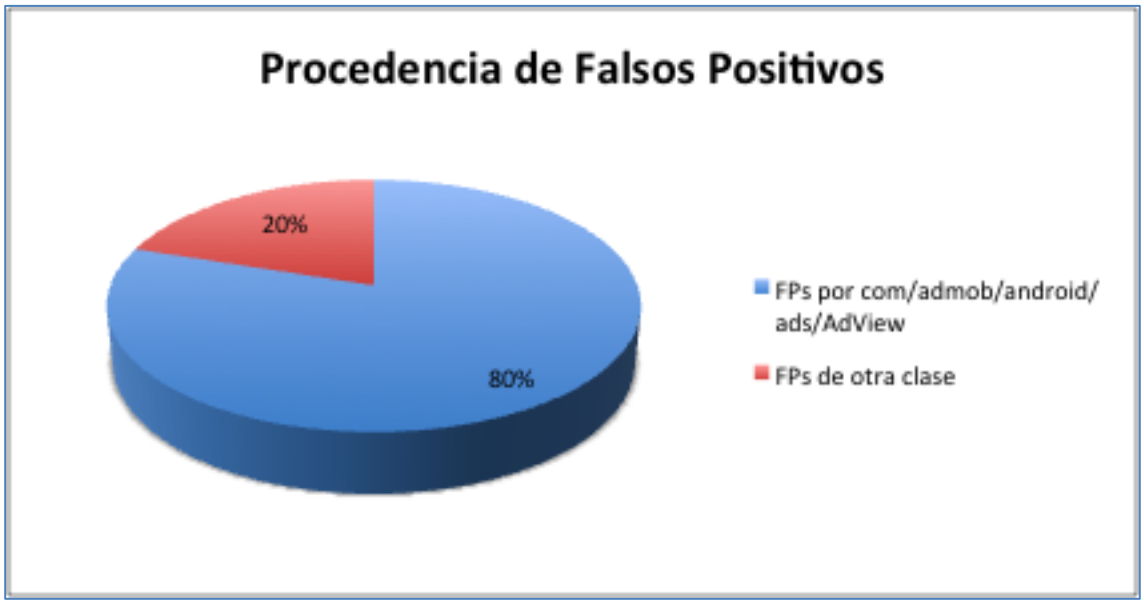

Figura 25. Gráfico de Procedencia de Falsos Positivos.

En cuanto a las posibles vulnerabilidades reales detectadas, también llamó la atención la repetición de una clase en concreto. Esta clase pertenece a un módulo publicitario desarrollado por la compañía Millennial Media. La inclusión de este módulo en las aplicaciones representa un total del 44% de las vulnerabilidades detectadas, y el 24% de las aplicaciones vulnerables. Estos datos se reflejan en los gráficos de la Figura 26 y la Figura 27.

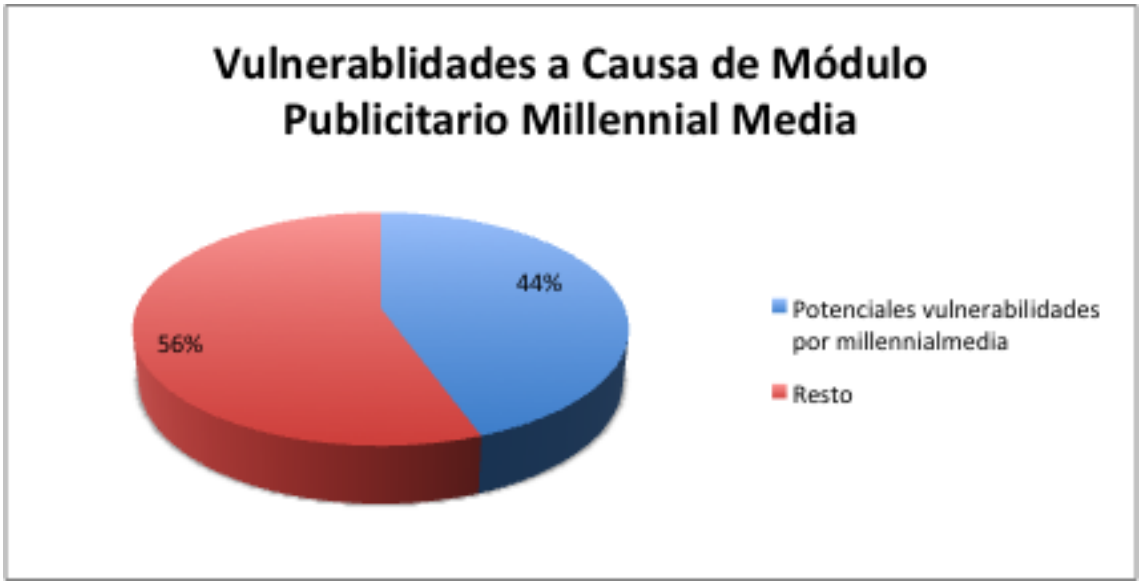

Figura 26. Gráfico de Vulnerabilidades Causadas por Millennial Media.

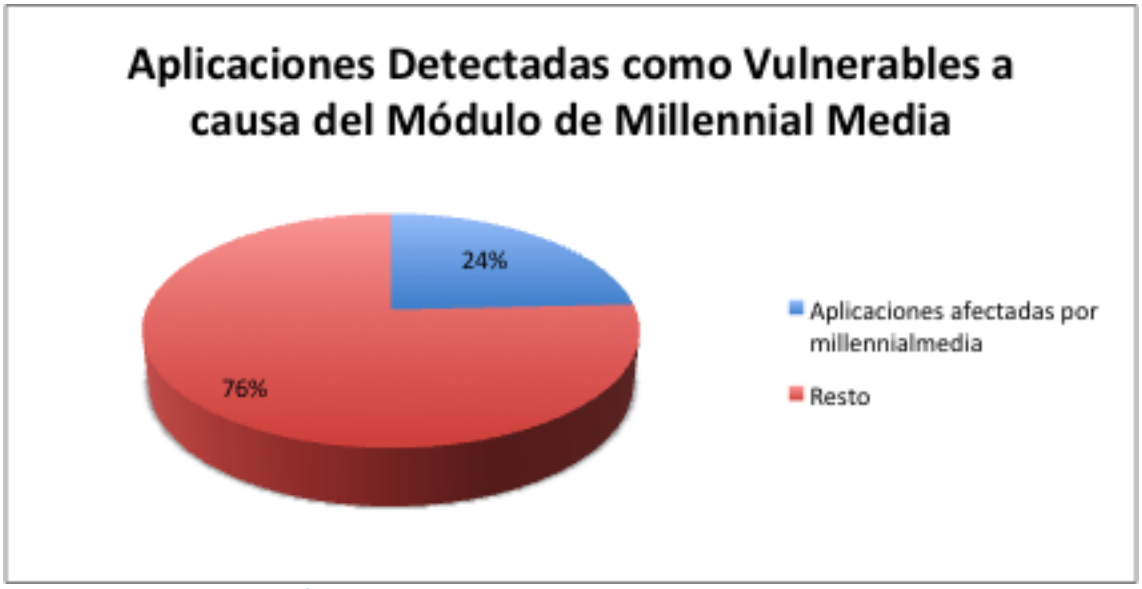

Figura 27. Gráfico de Aplicaciones Vulnerables a Causa de Millennial Media.

Si se divide entre aplicaciones legítimas y malware, los resultados que se obtienen tras el análisis es el siguiente.

Las aplicaciones legítimas potencialmente vulnerables suponen un 20% del total de las aplicaciones legítimas (Figura 28), y si se eliminan los falsos positivos, este porcentaje desciende hasta un 18% (Figura 29).

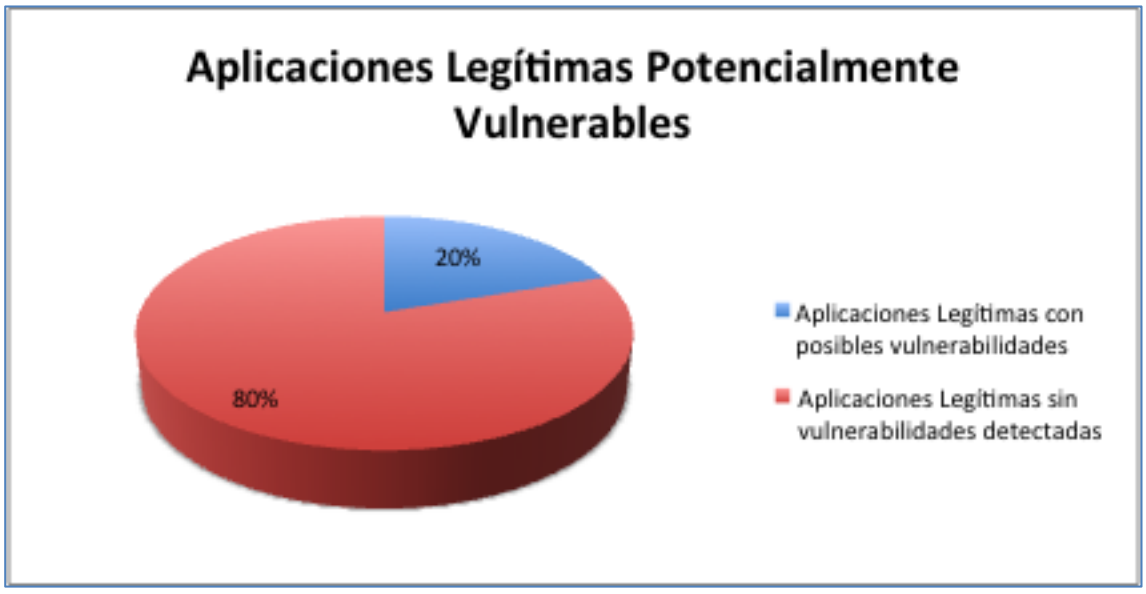

Figura 28. Gráfico de Aplicaciones Legítimas Potencialmente Vulnerables.

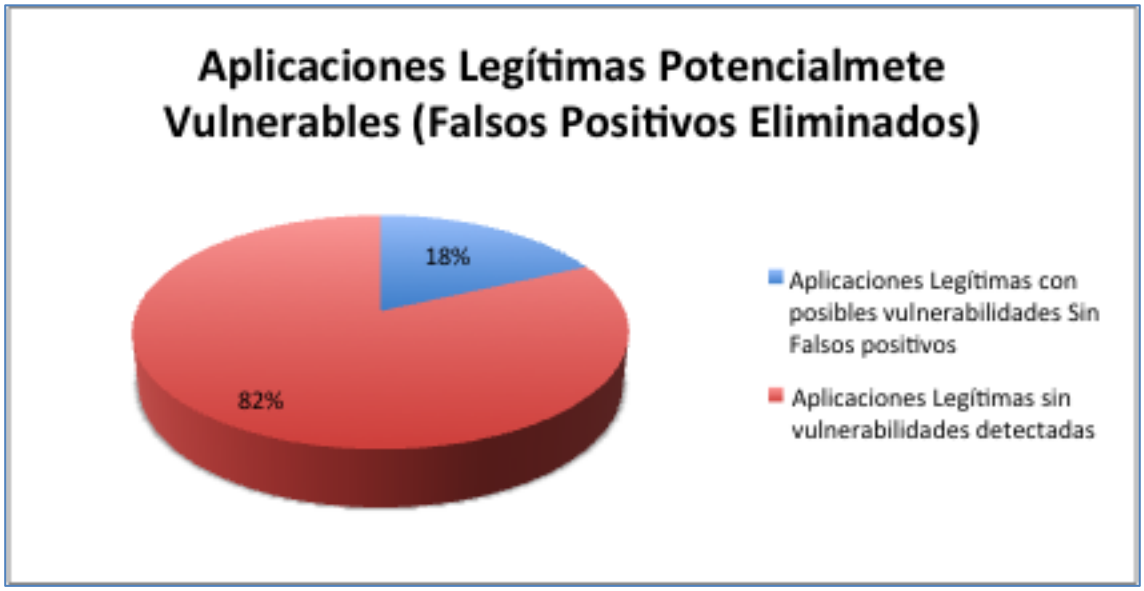

Figura 29. Gráfico de Aplicaciones Legítimas Potencialmente Vulnerables Reales.

En cuanto al malware, el resultado presentado ha sido del 22% aplicaciones malware potencialmente vulnerable, del total de las aplicaciones malware (Figura 30), y un 3% si se eliminan los falsos positivos(Figura 31).

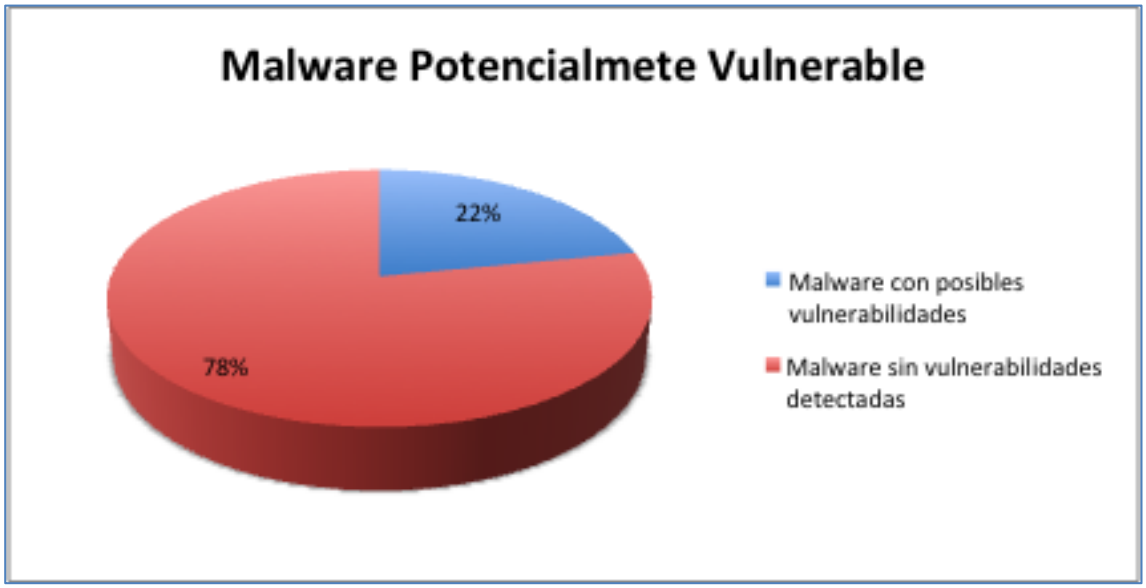

Figura 30. Gráfico de Malware Potencialmente Vulnerable.

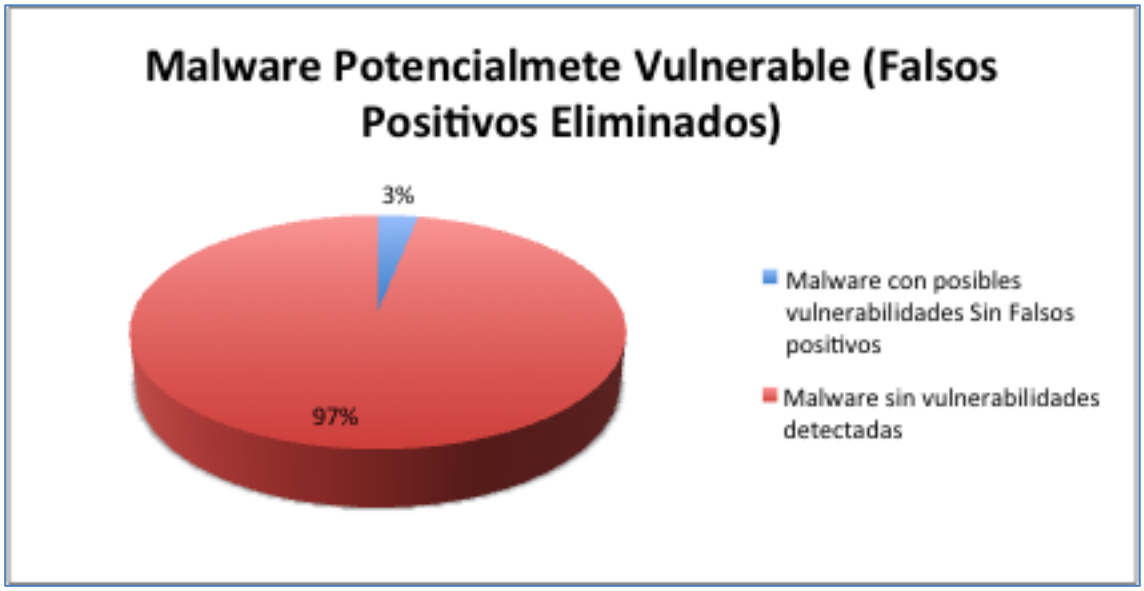

Figura 31. Gráfico de Malware Potencialmente Vulnerable Reales.

7.5. Conclusiones del estudio

La conclusión principal del estudio es:
"Pese a la gran cantidad de documentación que aporta la web de desarrolladores de Android, y la multitud de ejemplos disponibles en internet, sigue habiendo desarrolladores que realizan un mal uso de las funciones de Android para interactuar con bases de datos SQLite."

Lo preocupante de esta conclusión es que el porcentaje de aplicaciones que sufren de este problema, es demasiado elevado en opinión del autor. Al comienzo del proyecto se pensaba que quizás fuera difícil de encontrar estas posibles vulnerabilidades, sin embargo se ha podido ver que el 16% de las aplicaciones dispone de potenciales vulnerabilidades.

Otra gran conclusión del estudio es: que se debe tener cuidado, y valorar bien, la incorporación de módulos publicitarios a las aplicaciones que se desarrollen, ya que aunque directamente quizás no comprometan la seguridad de éstas, sí perjudicará gravemente la imagen de dicha aplicación y su compañía. Si esta aplicación se incluyese dentro del grupo de aplicaciones potencialmente vulnerables, la confianza en los usuarios hacia dicha aplicación se vería muy mermada.

En cuanto a los resultados por separado de las aplicaciones legitimas, y el malware analizado, señalar que la diferencia en el número de aplicaciones de la muestra de malware, ha podido influir en un resultado tan distinto. 18% de aplicaciones potencialmente vulnerables, frente al 1% de malware potencialmente vulnerable. Con un número similar de aplicaciones malware, se esperaría un resultado similar al de las aplicaciones legitimas.

Como última gran conclusión del estudio, se valora que el acierto a la hora de detectar posibles vulnerabilidades ha sido bastante alto, con una tasa de falsos

positivos de tan solo el 6%. Además, visto que gran parte de estos falsos positivos, provienen a raíz de un módulo publicitario en concreto, su corrección para versiones futuras sería muy sencillo. En resumen, para tratarse de su primera versión, el resultado del análisis es considerado como un éxito.

Capítulo 8. Conclusiones y líneas futuras

En este capítulo se exponen las conclusiones extraídas de la realización del proyecto, así como las líneas de futuro para continuación del mismo.

8.1. Conclusiones sobre el proyecto

Cuando se inició este proyecto, se definieron una serie de objetivos, que a día de la culminación del proyecto pueden darse por superados exitosamente. Estos resultados obtenidos con la finalización del proyecto han supuesto unas conclusiones más que satisfactorias y positivas.

Todos y cada uno los objetivos marcados han sido cumplidos, y algunos de ellos, con mejores resultados de los esperados.

- Destacar la aplicación Android, como una herramienta de concienciación para una programación segura, que podría ser utilizada como ejercicio de seguridad en la comunidad universitaria, y en los centros de estudios avanzados de informática.
- Resaltar la sencillez y buenos resultados de la aplicación de búsqueda de potenciales vulnerabilidades. Esta herramienta podría ser utilizada por las organizaciones y desarrolladores, para verificar la calidad del software que desarrollan, y también podría ser utilizado por pentesters (Pentesting, 2013), para auditar aplicaciones Android.
- Por último destacar también, el sorprendente resultado ofrecido por el estudio, que ha servido para demostrar, que las aplicaciones Android no son tan seguras como se esperaba al comienzo del proyecto. El resultado recogido por el estudio señaló que el 18% de las aplicaciones disponibles en el mercado oficial de Android, presentaba errores de seguridad en la programación de sus aplicaciones.

8.2. Conclusiones a nivel personal

La realización y conclusión de este proyecto ha representado a nivel personal, una situación de satisfacción enorme. Este proyecto simboliza por un lado la culminación de compromiso y esfuerzo que he realizado durante estos últimos cuatro años, desde que me matriculé en el Grado de Ingeniería Informática. Por el otro, este proyecto también simboliza mi inquietud por las nuevas tecnologías, y mi actitud de superación continua.

Podría haber elegido un proyecto más sencillo y con menos trabajo, pero decidí realizar un proyecto ambicioso tanto en temática como en tiempo. Según mis valores, cuándo uno se pone grandes metas, grande es la satisfacción al superarlas.

Mi curiosidad por la seguridad informática, me llevo a la decisión de realizar un proyecto en este campo. Gracias a esta decisión, mi experiencia y conocimientos en esta rama de la informática han sido enriquecedores, y han consolidado mi interés por la ciberseguridad.

Para concluir me gustaría señalar, que con este proyecto, se han fortalecido mis valores de responsabilidad, planificación y compromiso, hacia la realización de un trabajo bien hecho.

8.3. Líneas Futuras

En lo referente a la continuación del proyecto, los puntos por los que podría continuar su crecimiento serían los siguientes:

- Se podría aumentar el nº de ejercicios de la aplicación Android, y diversificar la temática de los ejercicios para cubrir un mayor campo de seguridad, siguiendo con el principio de concienciación para una programación segura de aplicaciones.
- Se podría crear una web dedicada al proyecto, dónde se expliquen los fallos de seguridad cometidos, y la manera de solventarlos.
- Se podría enriquecer la experiencia de usuario de la aplicación Android, añadiendo enlaces a dicha web, al superar los desafíos, y así conocer de forma inmediata el motivo de las vulnerabilidades y como solucionarlas.
- Se podría mejorar la detección de falsos positivos, para mejorar aun más la veracidad de los resultados de la aplicación de análisis de aplicaciones. Viendo los resultados del estudio, sería sencillo añadir una clausula que evitara los falsos positivos por los módulos publicitarios “admob-AdView”.
- Se podría modificar la aplicación de análisis, para que reciba de un fichero, las expresiones regulares que buscar. De esta manera se aumentaría el radio de búsqueda de vulnerabilidades a un mayor tipo de aplicaciones. Con esta modificación, ya no solo se tendrían por qué analizar aplicaciones Android, sino que se podría analizar el código fuente de cualquier tipo de software.

Por otro lado, se podría buscar la colaboración de universidades y centros de estudios avanzados en informática, para explotar las aplicaciones del proyecto, y continuar con su difusión y desarrollo. Con las modificaciones planteadas, la aplicación SQLinject podría ser utilizada como herramienta complementaria, para la formación de los alumnos en asignaturas de seguridad informática, y la aplicación de análisis, InjectSearch, podría ser empleada para la corrección de prácticas de dichas asignaturas.

En el ámbito profesional, como ya se ha mencionado, InjectSearch podría ser empleada para verificar la calidad de los productos desarrollados, así como de herramienta para el pentesting de aplicaciones.

ANEXO 1. Manual de Aplicaciones

En este anexo se desarrollan los manuales para el correcto uso de las aplicaciones. Se detallan paso a paso las instrucciones de ejecución para las operaciones disponibles en las distintas aplicaciones.

La primera aplicación a describir será la aplicación de concienciación, disponible para Android, y que recibe el nombre de SQLinject. Para esta aplicación, se dará una explicación de las pantallas principales, describiendo la utilidad de cada campo o botón.

La segunda aplicación a describir será la herramienta de análisis de aplicaciones Android, que recibe el nombre de InjectSearch. Para este programa, se dará una explicación de para qué, y cómo usar cada comando.

Manual de aplicación SQLinject (Aplicación Android)

En este apartado se detallará el modo de uso de la aplicación SQLinject. (Figura 32)

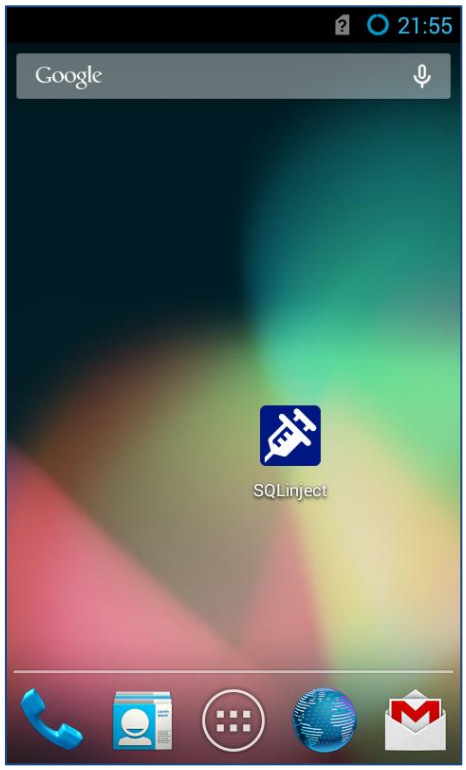

Figura 32. Aplicación SQLinject.

Pantalla principal

En la pantalla principal se encontrará el listado de ejercicios disponibles: “Login”, “Serach product”, “Enlist product”, “Chat 1”, y “Chat 2” (Figura 33).

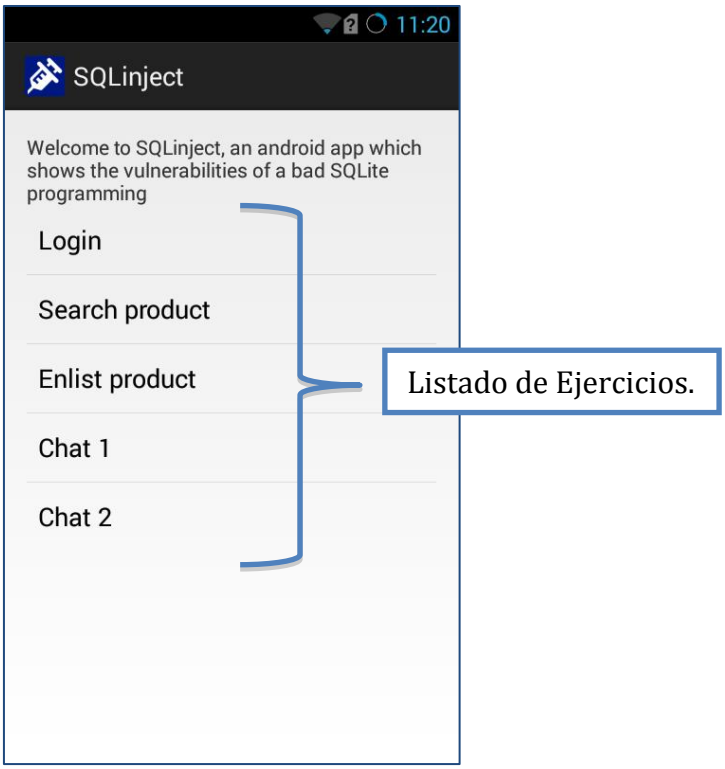

Figura 33. Listado de Ejercicios SQLinject.

Para acceder a uno de estos ejercicios, basta con pulsar sobre cualquiera de ellos, para que la aplicación abra dicha actividad.

Disponible en el menú principal, también se encuentra el botón para restablecer la base de datos. Este botón se encuentra oculto, y para que aparezca simplemente hay que pulsar el botón de opciones del dispositivo móvil (Figura 34).

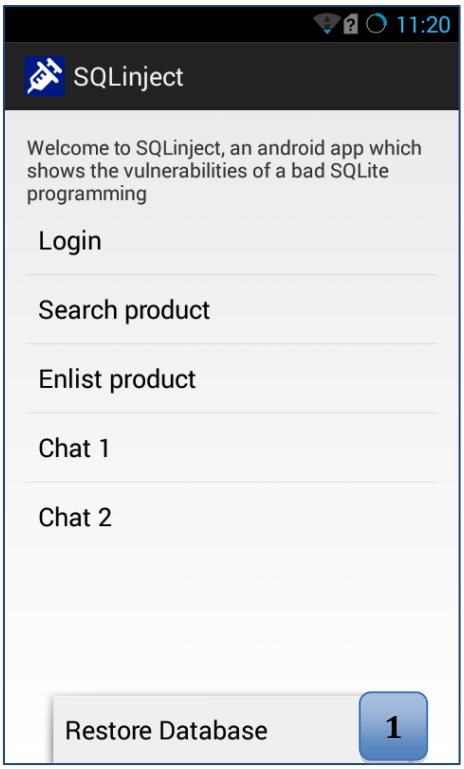

Figura 34. SQLinject - Botón Restore Database.

Pulsando el botón de "Restore Database", se reestablecerá la base de datos con la información inicial de la aplicación (1).

Ejercicio: Login

En la primera pantalla del ejercicio "Log in" se encuentra un formulario de acceso a un área restringida. Para completar el formulario se debe introducir el nombre de usuario en (1), y la contraseña correspondiente en (2). Para que el proceso de login sea correcto ambos campos deben estar cumplimentados (Figura 35).

Una vez rellenos los campos (1) y (2), se debe pulsar al botón "send" (3), para que se proceda con la operación de acceso.

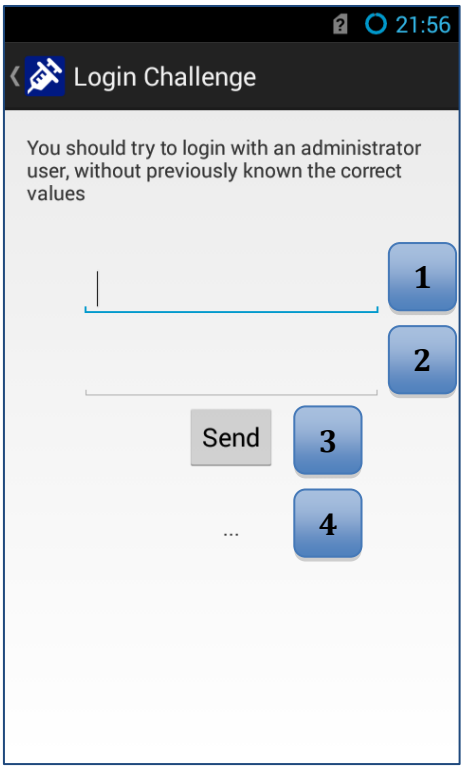

Figura 35. SQLinject - Ejercicio Login Pantalla 1.

Si en el proceso de acceso se produjera algún error, este sería notificado por una alerta emergente, y un mensaje en la sección (4).

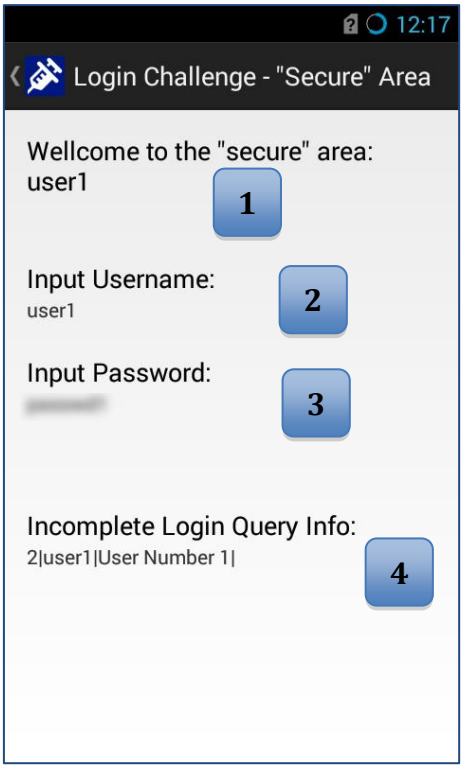

Figura 36. SQLinject - Ejercicio Login Pantalla 2.

Una vez se realiza un acceso correcto, se accede a la pantalla representada en la Figura 36. Dado que es una aplicación formativa, la información proporcionada en esta pantalla, aporta información útil al "alumno". En el campo (1) se da la bienvenida al usuario, en el campo (2) se imprime la cadena introducida en el campo usuario de la ventana anterior, en le campo (3) se muestra la cadena introducida en el campo de contraseña de la ventana anterior, por último en el campo (4) se muestra al "alumno", el resultado de la query lanzada.

Ejercicio: Search product

El ejercicio "Search product", solo se compone de una pantalla desde la que realizar el ejercicio. Esta pantalla solo se compone de tres elementos (Figura 37). En (1) se encuentra el campo de texto para la búsqueda del producto, en este campo deberá escribirse información relevante a la referencia del producto, el elemento (2), representa el botón "Search", que habrá que pulsar para solicitar la operación de búsqueda. En esta ocasión el campo de búsqueda puede encontrarse vacío, en tal caso, el resultado de la búsqueda serán todos los productos almacenados. Por último, el elemento (3), compone el resultado de la consulta.

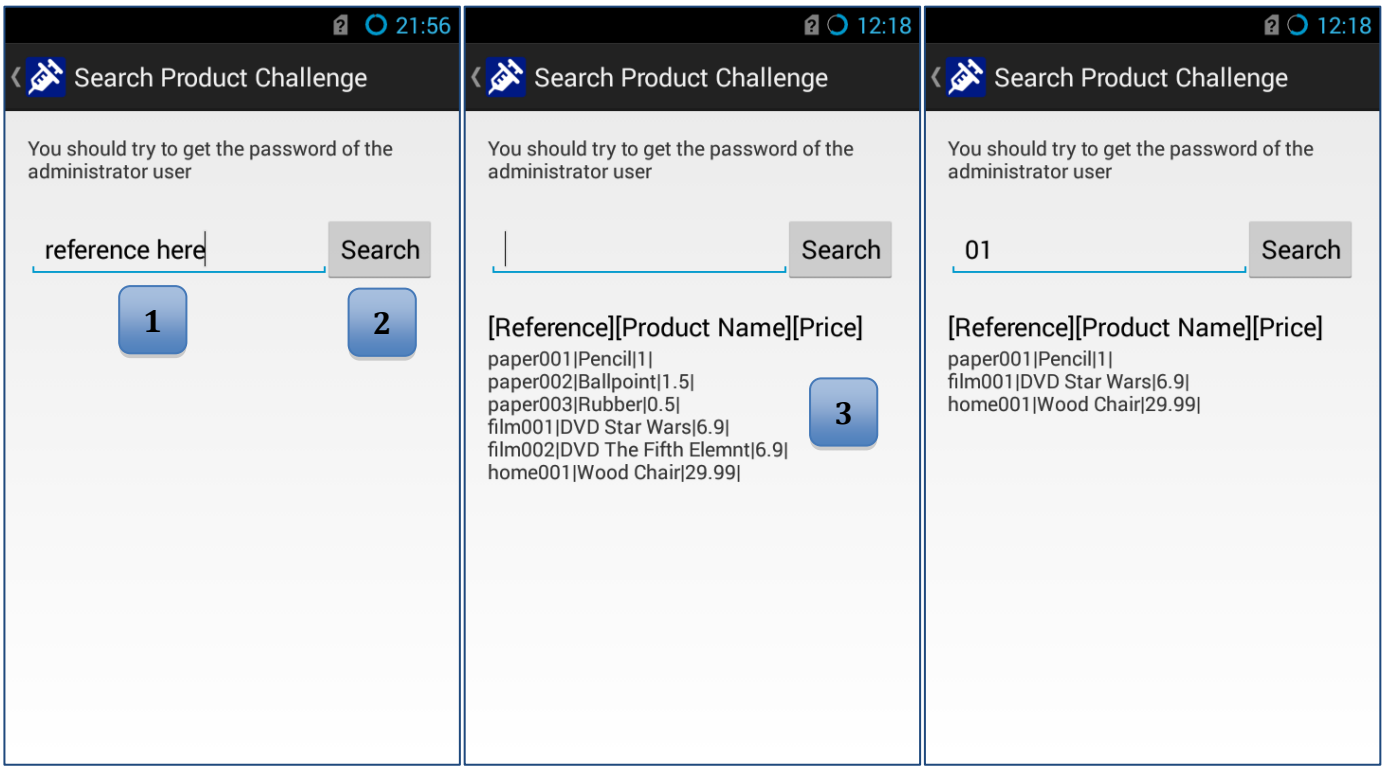

Figura 37. SQLinject - Ejercicio Search Product.

Tal como se observa en la Figura 37, la cadena introducida en el campo de búsqueda, retorna todos los elementos que contengan dicha cadena en su referencia

Ejercicio Enlist product

La actividad "Enlist product", está compuesta de dos ventanas. La primera de las ventanas corresponde a un formulario de alta de producto, en la que debe

rellenarse todos los campos de texto, para que al pulsar el botón se realice la operación correctamente (Figura 38).

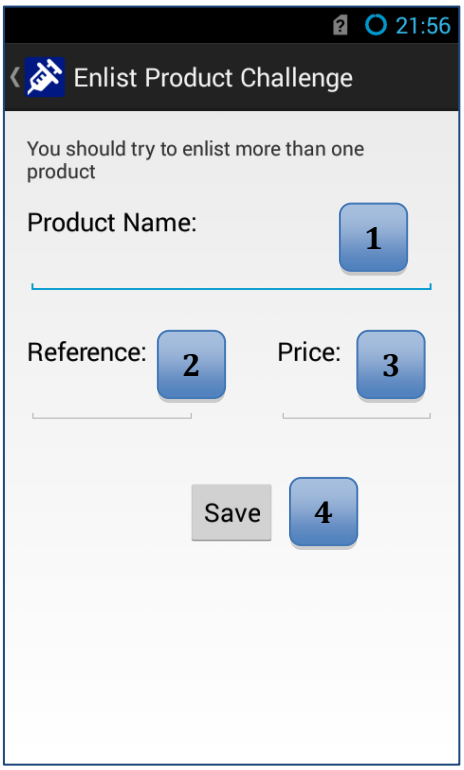

Figura 38. SQLinject - Ejercicio Enlist Product Pantalla 1.

En el campo (1) deberá introducirse el nombre del producto, en el campo (2) se deberá introducir la referencia del producto, y en el campo (3) se escribirá el valor de dicho producto. El botón (4) se deberá pulsar cuando todos los campos estén cumplimentados. En caso contrario se notificará al usuario, que debe rellenar todos los campos.

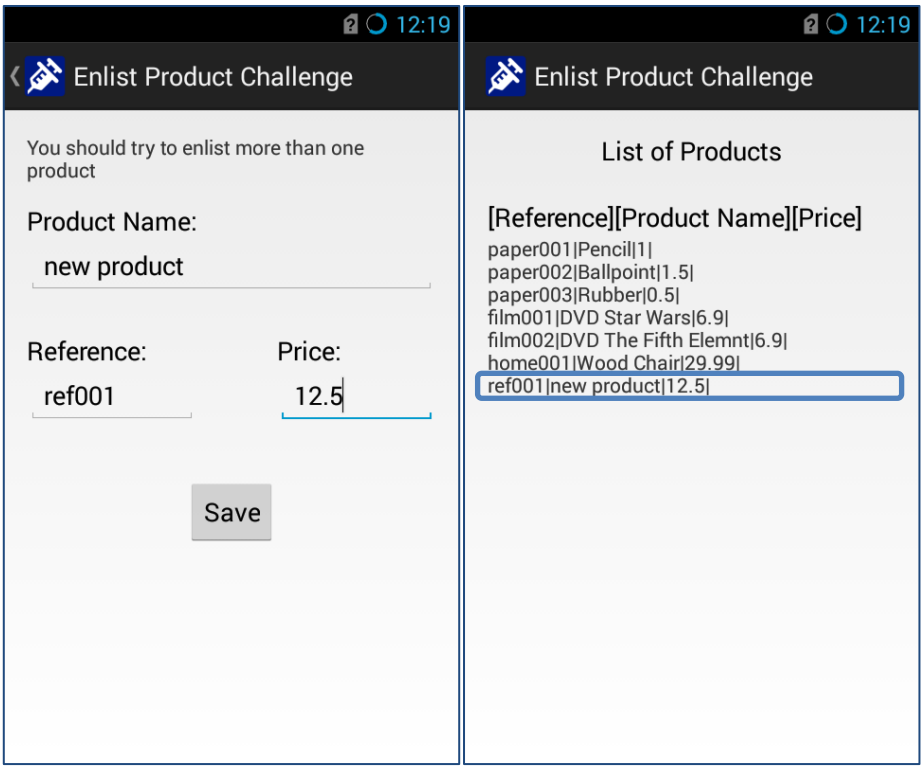

Figura 39. SQLinject - Ejercicio Enlist Product Pantallas 1 y 2.

Introducidos todos los datos de un nuevo producto, y pulsado el botón "Save", se abrirá la segunda ventana de la actividad, en la que listarán todos los productos de la base de datos, con el recién añadido, al final de la lista (Figura 39).

Ejercicio Chat 1

En el ejercicio de Chat 1 encontramos una única ventana dividida en dos pestañas. Una de las pestañas representa la ventana de chat del usuario, mientras que la otra representa la ventana de chat de su interlocutora Alice.

Esta ventana está dividida en los siguientes elementos, (1) pestañas para elegir el chat de usuario, o el chat de Alice. Basta con pulsar sobre cualquiera de ellas para posicionarse en el chat elegido. (2) Es el área donde se irán cargando los mensajes enviados. El campo de redacción del mensaje, se identifica como el elemento (3), y el botón para enviar el mensaje con el elemento (4) (Figura 40).

Para que un mensaje sea enviado al pulsar el botón de enviar, el campo de redacción del mensaje debe tener algo escrito, en caso contario, no será enviado mensaje alguno.

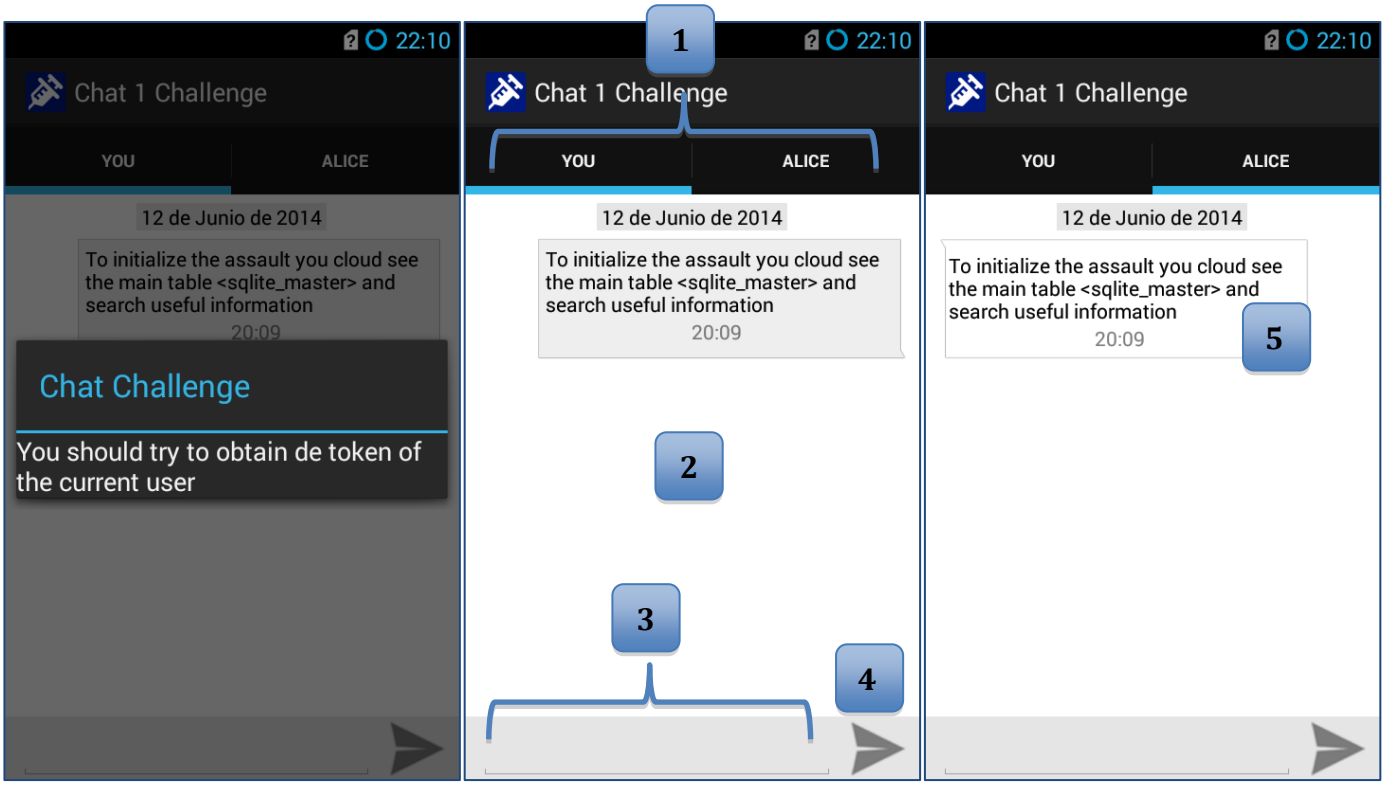

Figura 40. SQLinject - Ejercicio Chat 1.

El elemento (5) representa un mensaje de la conversación, y como puede observarse, dependiendo de quién sea el emisor tomará la posición a la izquierda o a la derecha del área de mensajes. Cuando el mensaje haya sido enviado por el propietario del chat abierto (pestaña de usuario, o pestaña de Alice), el mensaje se mostrará en el lateral derecho del área de mensajes, en caso contrario, el mensaje se postrará en el lateral izquierdo.

A continuación se presenta una secuencia de intercambio de mensajes en los que se ve de forma clara el procedimiento de uso (Figura 41, Figura 42, y Figura 43).

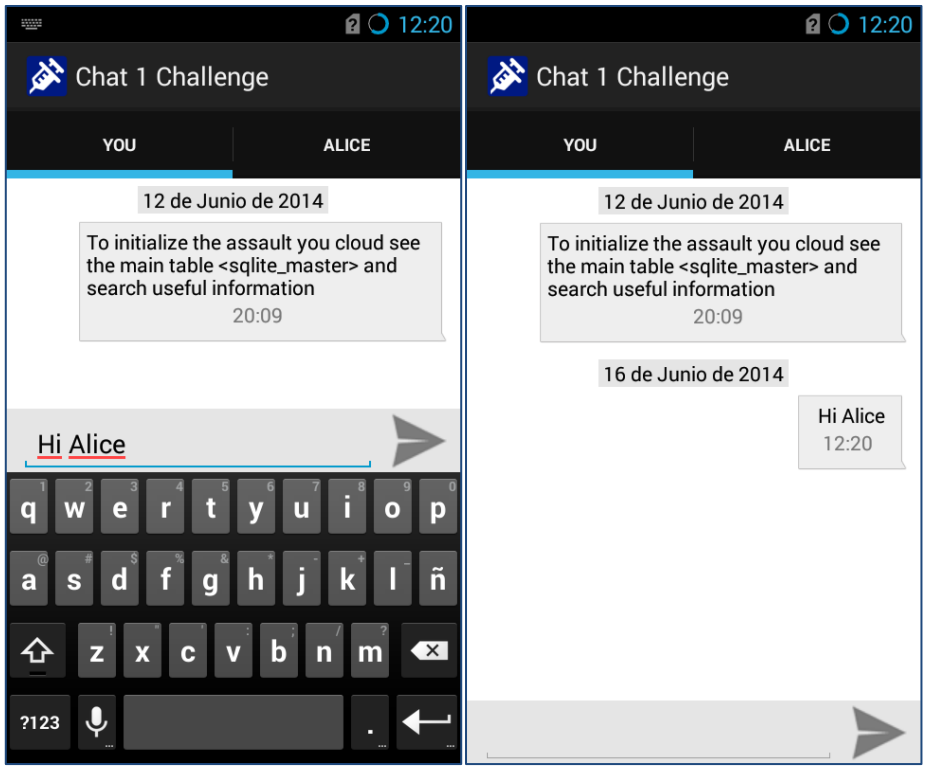

Figura 41. SQLinject - Ejercicio Chat 1, Usuario envía mensaje.

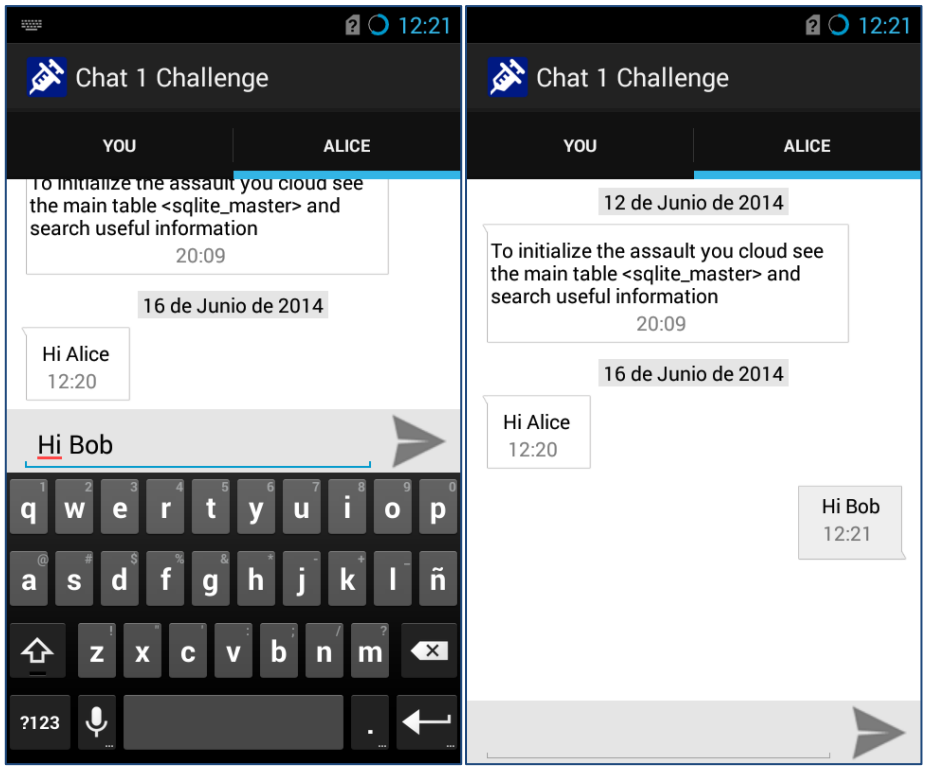

Figura 42. SQLinject - Ejercicio Chat 1, Alice contesta al mensaje.

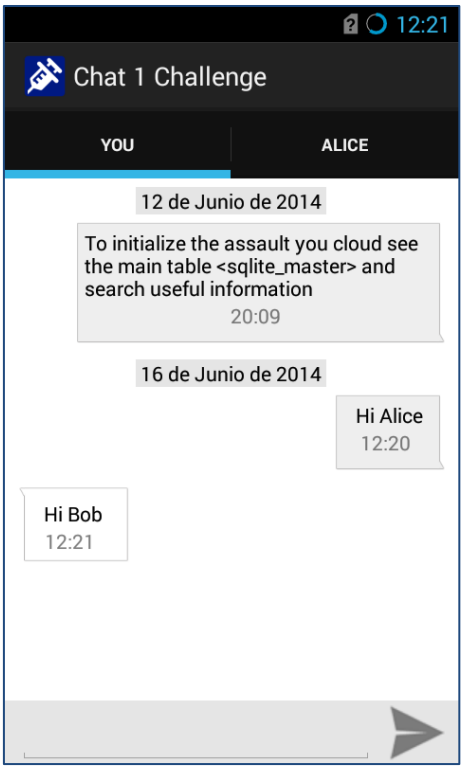

Figura 43. SQLinject - Ejercicio Chat 1, Usuario recibe mensaje de Alice.

Ejercicio Chat 2

En el ejercicio de Chat 2 encontramos una única ventana dividida en dos pestañas. Una de las pestañas representa la ventana de chat del usuario, mientras que la otra representa la ventana de chat de su interlocutora Alice.

Esta ventana está dividida en los siguientes elementos, (1) pestañas para elegir el chat de usuario, o el chat de Alice. Basta con pulsar sobre cualquiera de ellas para posicionarse en el chat elegido. (2) Es el área donde se irán cargando los mensajes enviados. El campo de redacción del mensaje, se identifica como el elemento (3), y el botón para enviar el mensaje con el elemento (4) (Figura 44).

Para que un mensaje sea enviado al pulsar el botón de enviar, el campo de redacción del mensaje debe tener algo escrito, en caso contrario, no será enviado mensaje alguno.

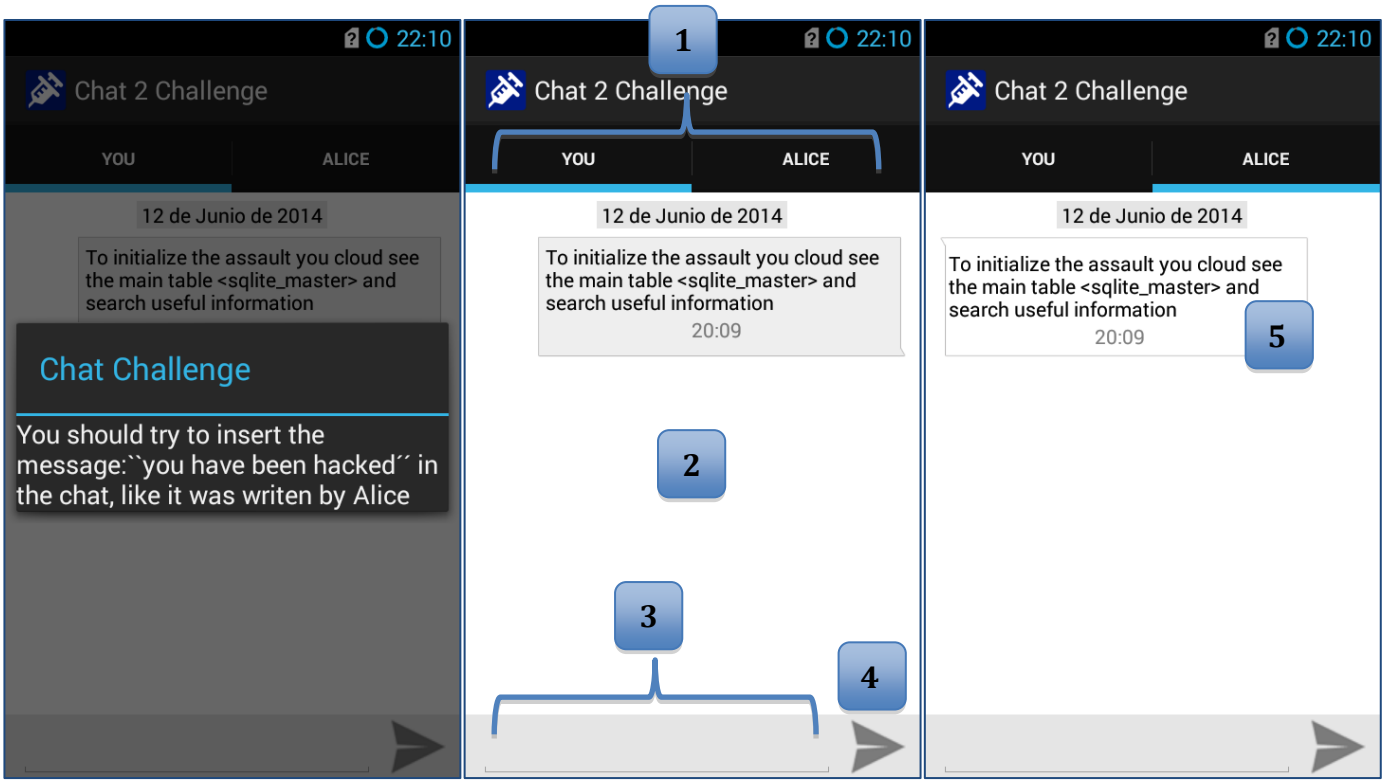

Figura 44. SQLinject - Ejercicio Chat 2.

El elemento (5) representa un mensaje de la conversación, y como puede observarse, dependiendo de quién sea el emisor tomará la posición a la izquierda o a la derecha del área de mensajes. Cuando el mensaje haya sido enviado por el propietario del chat abierto (pestaña de usuario, o pestaña de Alice), el mensaje se mostrará en el lateral derecho del área de mensajes, en caso contrario, el mensaje se postrará en el lateral izquierdo.

A continuación se presenta una secuencia de intercambio de mensajes en los que se ve de forma clara el procedimiento de uso (Figura 45, y Figura 46).

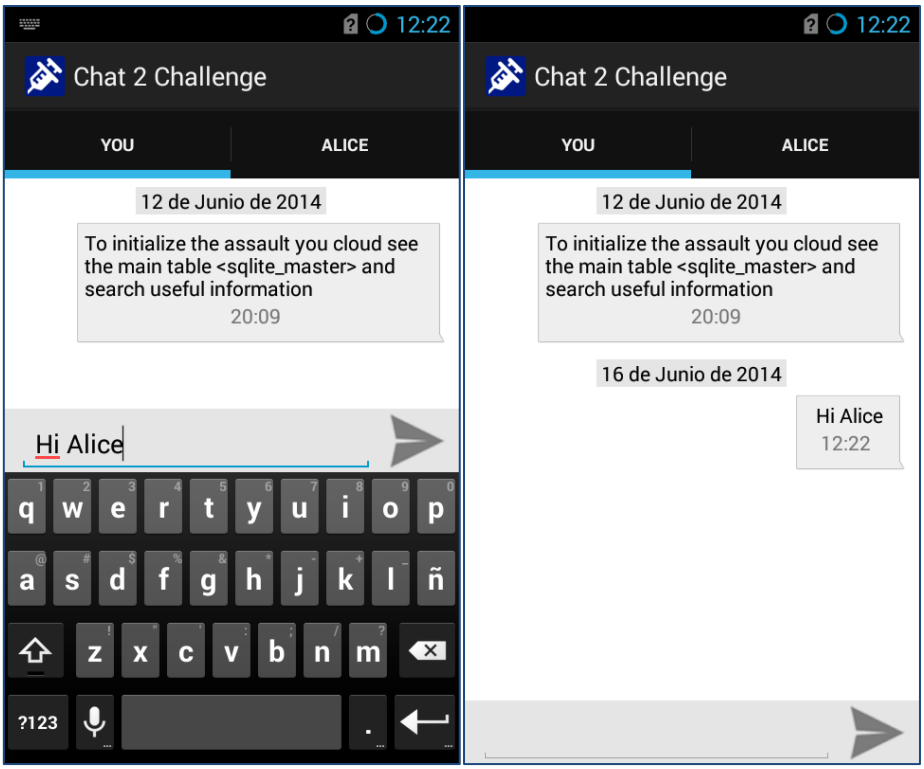

Figura 45. SQLinject - Ejercicio Chat 2, Usuario envía mensaje.

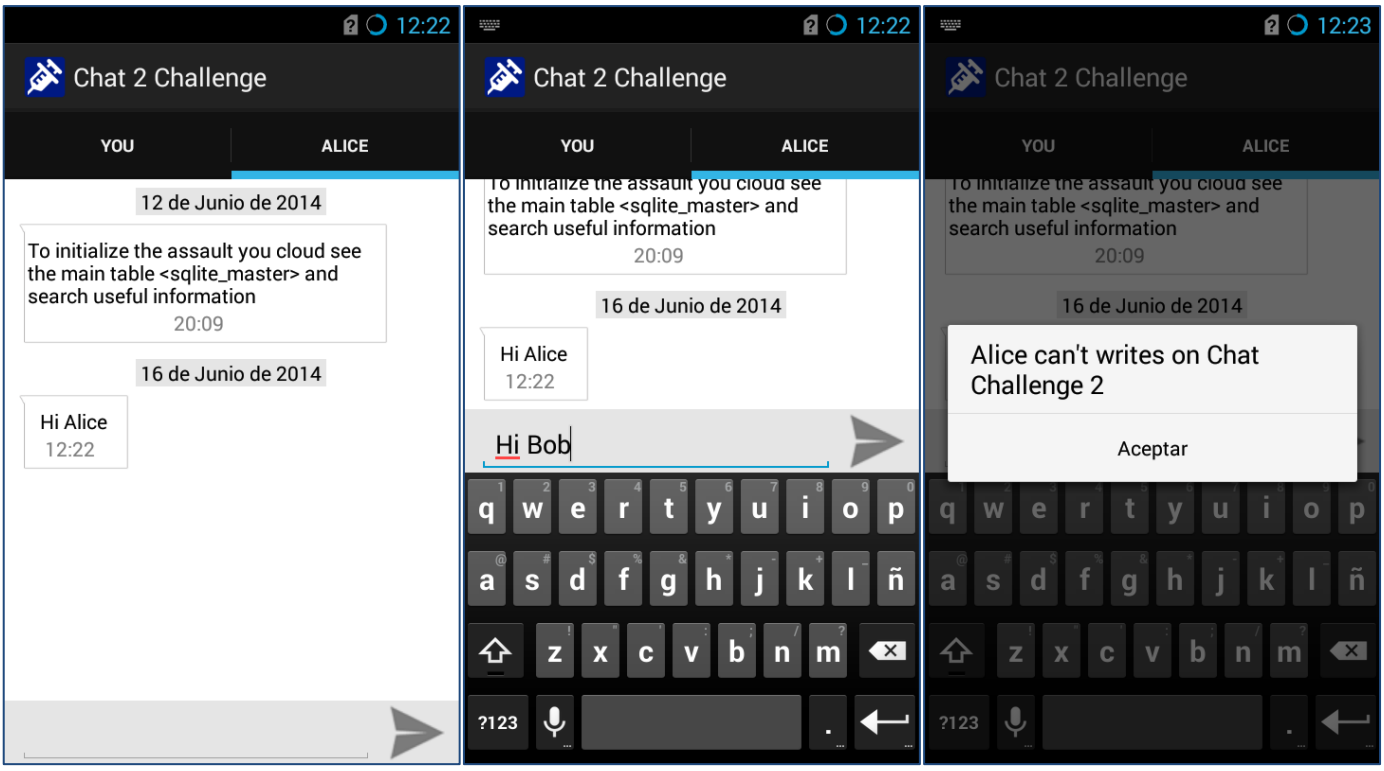

Figura 46. SQLinject - Ejercicio Chat 2, Alice recibe el mensaje e intenta contestar.

Como puede verse en esta ocasión según la Figura 46, desde la ventana de chat de Alice no está permitido mandar mensajes, esto es debido a que el objetivo en este

ejercicio es hacerse pasar por Alice, mandando un mensaje como si fuera ella quien lo envía.

Manual de aplicación InjectSearch (Script de Análisis)

En este apartado se detallará el modo de uso de la aplicación de consola InjectSearch.

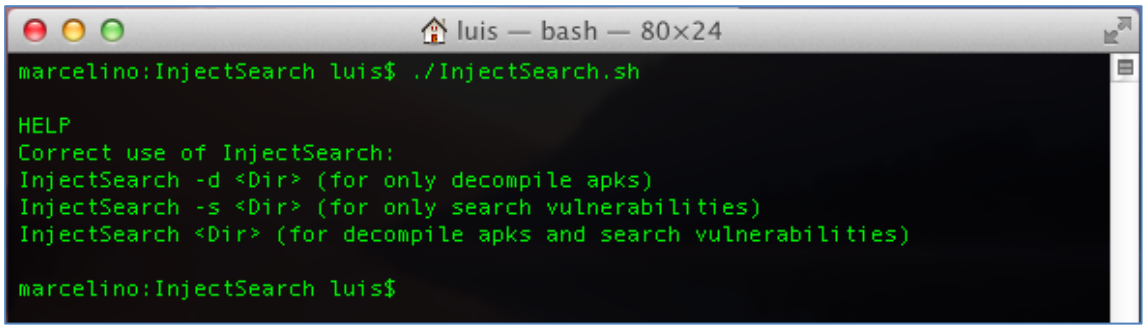

Figura 47. Aplicación InjectSearch.

Opciones disponibles

La herramienta InjectSearch, dispone de tres opciones de utilización como se refleja en la Figura 47. Estas opciones de uso son la opción de decompilación, la opción de análisis, o por último la opción automática de decompilación y análisis.

El uso correcto de la herramienta es el reflejado por la Figura 47. Si se quiere decompilar, la herramienta debe ejecutarse con la opción –d, y recibiendo el directorio donde se almacenan las aplicaciones Android en formato apk. Si lo que se quiere es analizar las aplicaciones, que fueron decompiladas en algún momento anterior, la herramienta debe ejecutarse con la opción –s, y recibiendo el directorio donde se almacenan dichas aplicaciones decompiladas a analizar. Por último, si lo que se desea es realizar ambas tareas de forma automática, la herramienta debe ejecutarse recibiendo únicamente la dirección donde se encuentren las aplicaciones Android en formato apk.

Si la herramienta detecta que se quiere ejecutar con distintos parámetros a los señalados, mostrará la ayuda expuesta en la Figura 47.

Lanzar decompilación

Tal como se ha indicado en el apartado anterior, para lanzar la herramienta en modo decompilación, la aplicación debe ejecutarse con la opción –d, y recibiendo el directorio donde se almacenan las aplicaciones Android en formato apk (Figura 48).

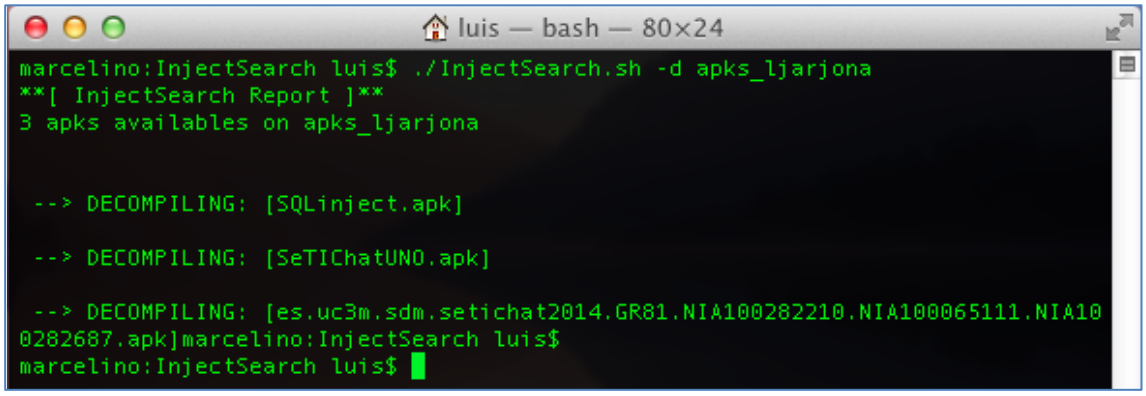

Figura 48. InjectSearch - Opción de decompilación 1.

El resultado a dicha operación genera un informe tal como reflejan la Figura 49, y la Figura 50.

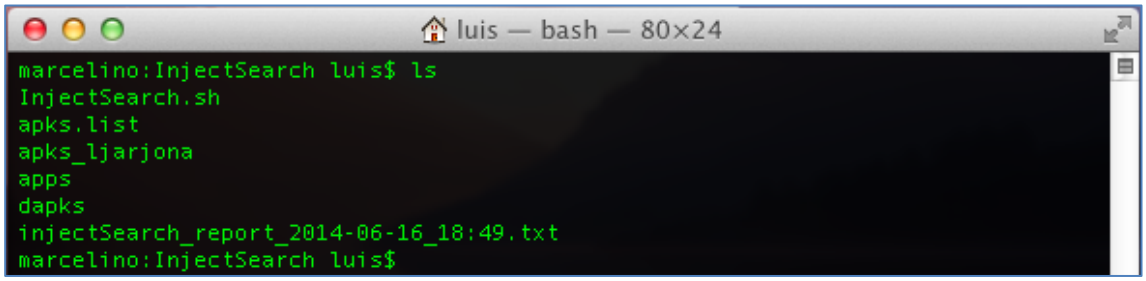

Figura 49. InjectSearch - Opción de decompilación 2.

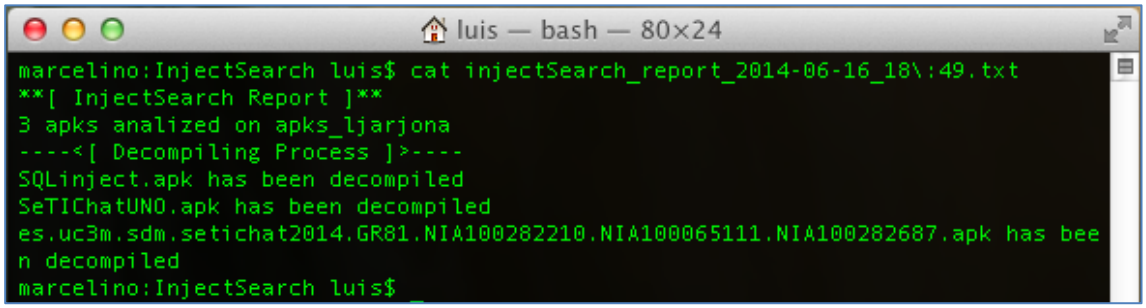

Figura 50. InjectSearch - Opción de decompilación 3.

Lanza análisis

Tal como se ha indicado en el apartado anterior, para lanzar la herramienta en modo decompilación, la aplicación debe ejecutarse con la opción –s, y recibiendo el directorio donde se almacenan dichas aplicaciones decompiladas a analizar (Figura 51).

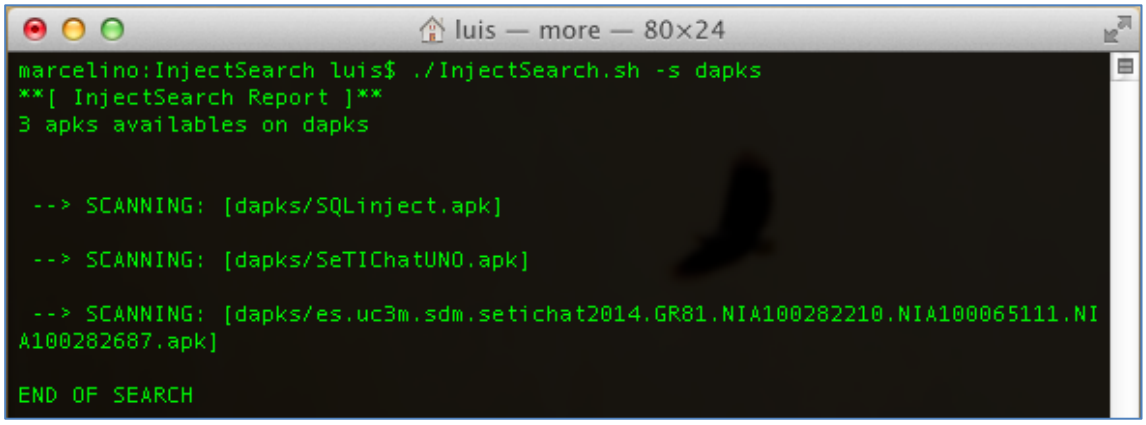

Figura 51. InjectSearch - Opción de análisis 1.

El resultado de dicho análisis es el informe presentado al finalizar las operaciones tal como se refleja en la Figura 52.

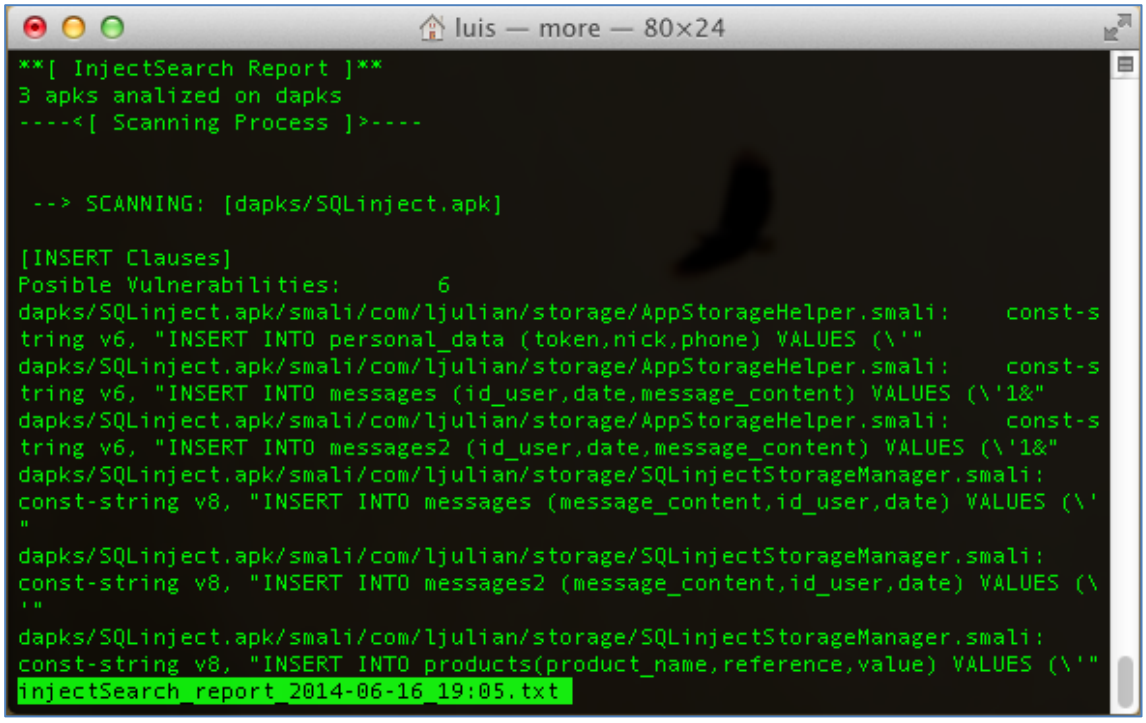

Figura 52. InjectSearch - Opción de análisis 2.

Lanzar decompilación y análisis

Para realizar la decompilación y análisis automático, la herramienta dispone de esta tercera opción que se ejecuta cuando solo se le indica la ruta con las aplicaciones Android en formato apk (Figura 53).

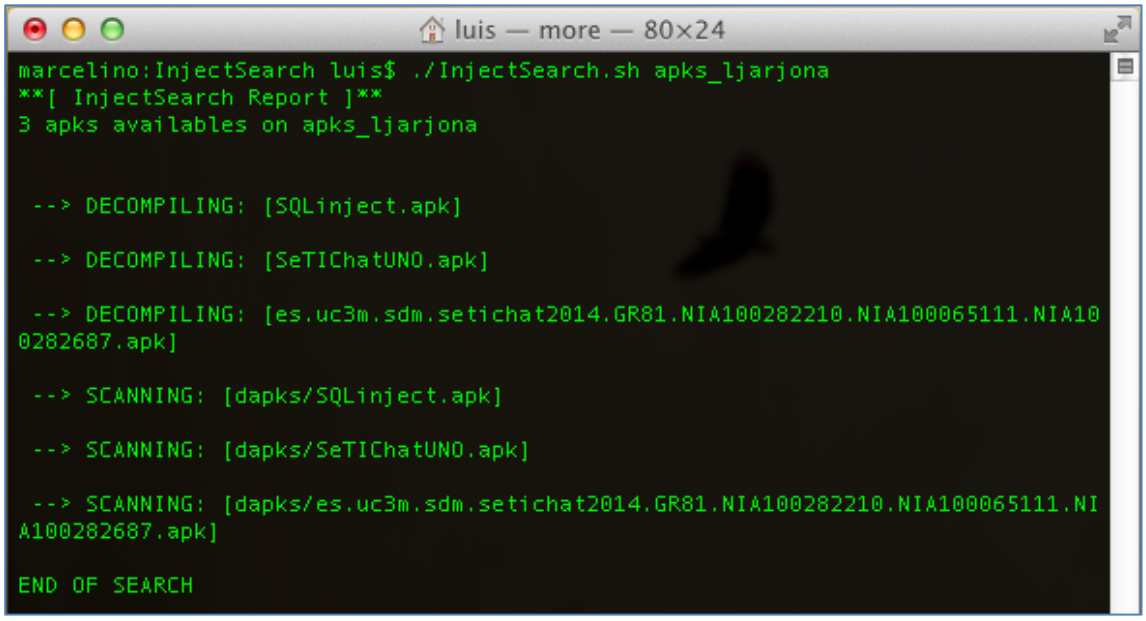

Figura 53. InjectSearch - Opción automática 1.

El resultado de dichas operaciones es el informe presentado al finalizar dicho proceso tal como se refleja en la Figura 54.

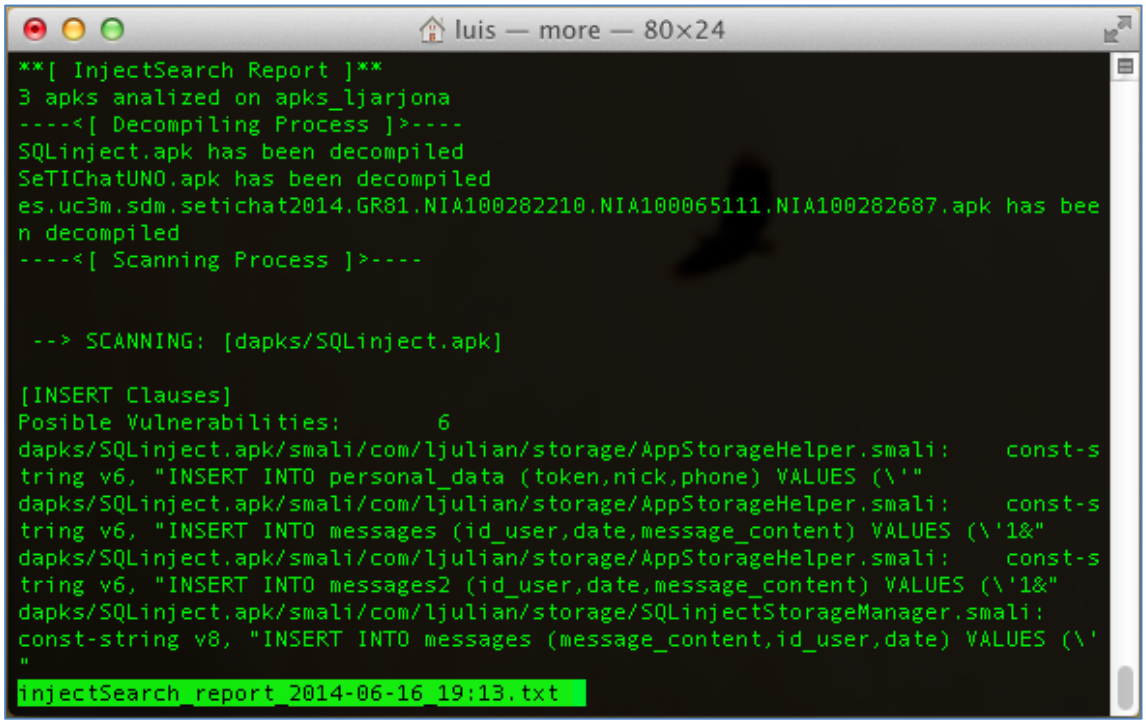

Figura 54. InjectSearch - Opción automática 2.

ANEXO 2. Listado de Aplicaciones Analizadas en el Estudio

El listado de categorías y aplicaciones analizadas ha sido:

- APP_WALLPAPER
  - com.androidwasabi.livewallpaper.dandelion_21.apk
  - com.androidwasabi.livewallpaper.waterdrop_13.apk
  - com.badoo.mobile_49.apk
  - com.google.android.apps.maps_612001402.apk
  - com.google.android.apps.maps_614001102.apk
  - com.picsart.studio_46.apk
  - slide.photoWallpaper_13.apk
- APP_WIDGETS
  - com.facebook.katana_48315.apk
  - com.facebook.katana_79836.apk
  - com.google.android.apps.maps_612001402.apk
  - com.google.android.apps.plus_311328793.apk
  - com.google.android.gm_176.apk
  - com.google.android.music_811.apk
  - com.google.android.youtube_4123.apk
- BOOKS_AND_REFERENCE
  - com.google.android.apps.books_2630.apk
  - com.google.android.apps.books_2640.apk
  - com.google.android.stardroid_1112.apk
  - it.seatpg.paginebianche_4.apk
  - livio.pack.lang.it_IT_20.apk
  - org.wikipedia_17.apk
  - org.wikipedia_23.apk
- BUSINESS
  - cn.wps.moffice_eng_44.apk
  - cn.wps.moffice_eng_47.apk
  - com.dataviz.docstogo_1292.apk
  - com.dynamixsoftware.printershare_154.apk
  - com.mobisystems.office_966.apk
  - com.rhmsoft.fm_51.apk
  - com.rhmsoft.fm_55.apk
- COMICS
  - com.androidity.wallpaper.funny2_2.apk
  - com.anjokes.apps.jokes.it_11.apk
  - com.darkhorse.digital_11.apk
  - com.keyspice.photocomics_21.apk
  - com.troll.face_10.apk
  - com.wImmaginieMemesDivertenti_1347624192.apk
- COMMUNICATION
  - com.facebook.orca_49466.apk
  - com.google.android.gm_176.apk
  - com.skype.raider_34144571.apk

    - com.viber.voip_19.apk
    - com.whatsapp_35310.apk
    - com.whatsapp_37326.apk
- EDUCATION
    - com.absoplex.patente.auto_3.apk
    - com.babbel.mobile.android.en_17.apk
    - com.idalmedia.android.timetable_31.apk
    - com.studiociriello.quiz.patente.autofree_10.apk
    - com.studiociriello.quiz.patente.autofree_12.apk
    - eu.appsolutelyapps.quizpatente_2.apk
    - tobi.tools.timetable_180.apk
- ENTERTAINMENT
    - com.Hace.Iphone5Screen_3.apk
    - com.mdb.android.fakeiphone5_5.apk
    - com.outfit7.talkinggingerfree_10.apk
    - com.softwares.daicazzo_121.apk
    - com.wifi.hacker.cracker_88889.apk
    - org.goldennuggetapps.simpledl_29.apk
    - org.goldennuggetapps.simpledl_31.apk
- FINANCE
    - com.fineco.it_2.apk
    - com.noverca.mbanking_15.apk
    - com.paypal.android.p2pmobile_28.apk
    - com.paypal.android.p2pmobile_29.apk
    - com.unicredit_6.apk
    - posteitaliane.posteapp.apppostepay_7.apk
- GAME
    - com.game.BubbleShooter_7.apk
    - com.gamestar.pianoperfect_604.apk
    - com.halfbrick.fruitninjafree_1603.apk
    - com.kiloo.subwaysurf_7.apk
    - com.rovio.angrybirdsstarwars.ads.iap_1010.apk
    - com.topfreegames.bikeracefreeworld_2012092619.apk
- HEALTH_AND_FITNESS
    - com.google.android.maps.mytracks_56.apk
    - com.macropinch.hydra.android_24.apk
    - com.popularapp.periodcalendar_18.apk
    - com.popularapp.periodcalendar_20.apk
    - com.runtastic.android_41.apk
    - net.cachapa.libra_272.apk
- LIBRARIES_AND_DEMO
    - com.levelup.bw.forecast_5.apk
    - com.svox.langpack.installer_13.apk
    - io.vov.vitamio.v6vfp_7.apk
    - io.vov.vitamio_7.apk
    - org.kde.necessitas.ministro_7.apk
- LIFESTYLE
    - com.acotel.astri_8.apk
    - com.ikea.catalogue.android_17.apk

    - com.opentecheng.paginegialle.dream_41.apk
    - it.banzai.media.gzricette_8.apk
    - it.h3g.areaclienti3_130.apk
    - it.h3g.areaclienti3_141.apk
- MEDIA_AND_VIDEO
    - com.google.android.youtube_4123.apk
    - com.iuculano.fplayer_27.apk
    - com.mxtech.videoplayer.ad_40.apk
    - com.mxtech.videoplayer.ad_44.apk
    - com.utorrent.client_15.apk
    - it.telecomitalia.cubovision_22.apk
    - tr.gen.hyper.tv.live.italy_1.apk
- MEDICAL
    - appinventor.ai_kosma822.IpnosiRegressivaLite_7.apk
    - com.eperto.app.smartpharma_3541.apk
    - com.remind4u2.sounds.of.rain_2.apk
    - it.visiant.farmacia.turno_4.apk
    - it.visiant.farmacie_14.apk
    - mm.app.formulazioni_1.apk
- MUSIC_AND_AUDIO
    - com.google.android.music_811.apk
    - com.melodis.midomiMusicIdentifier.freemium_10517.apk
    - com.pack.carman_232.apk
    - com.shazam.android_76414.apk
    - com.shazam.android_76601.apk
    - hr.podlanica_17.apk
    - tunein.player_28.apk
- NEWS_AND_MAGAZINES
    - com.google.android.apps.currents_121741107.apk
    - com.jappit.android.televideo_6.apk
    - com.tilab_8.apk
    - flipboard.app_170.apk
    - it.froggy.tgcom_11.apk
    - it.froggy.tgcom_13.apk
- PERSONALIZATION
    - com.androidwasabi.livewallpaper.dandelion_21.apk
    - com.androidwasabi.livewallpaper.waterdrop_13.apk
    - com.gau.go.launcherex_126.apk
    - com.gau.go.launcherex_142.apk
    - com.srsdev.wallpapers_319.apk
    - com.whatsapp.wallpaper_2.apk
    - slide.photoWallpaper_13.apk
- PHOTOGRAPHY
    - com.adobe.psmobile_9.apk
    - com.appspot.swisscodemonkeys.camerafx_6.apk
    - com.bluecode.photo.space.effects.fx_7.apk
    - com.fingersoft.cartooncamera_13.apk
    - com.picsart.studio_46.apk
    - ymst.android.fxcamera_2303.apk

    - ymst.android.fxcamera_2500.apk
- PRODUCTIVITY
    - com.adobe.reader_67615.apk
    - com.adobe.reader_68816.apk
    - com.bazaar.installer_262.apk
    - com.dropbox.android_211000.apk
    - com.estrongs.android.pop_99.apk
    - com.gau.go.launcherex.language.it_1.apk
    - com.google.android.apps.docs_1100429.apk
- SHOPPING
    - com.ebay.annunci_15.apk
    - com.ebay.mobile_26.apk
    - com.ebay.mobile_27.apk
    - com.google.zxing.client.android_87.apk
    - com.groupon_2414.apk
    - uk.amazon.mShop.android_33.apk
- SOCIAL
    - com.facebook.katana_48315.apk
    - com.facebook.katana_79836.apk
    - com.google.android.apps.plus_311328793.apk
    - com.instagram.android_310.apk
    - com.instagram.android_320.apk
    - com.sgiggle.production_38.apk
    - com.twitter.android_175.apk
- SPORTS
    - com.eurosport_59.apk
    - com.jappit.calcio_36.apk
    - com.jappit.calcio_41.apk
    - com.livescore_7.apk
    - com.livescore_9.apk
    - it.froggymedia.sportmediaset_16.apk
    - it.telecomitalia.calcio_5.apk
- TOOLS
    - com.avast.android.mobilesecurity_2880.apk
    - com.devuni.flashlight_137.apk
    - com.devuni.flashlight_138.apk
    - com.google.android.apps.translate_122.apk
    - com.google.android.apps.translate_141.apk
    - com.google.android.voicesearch_214.apk
    - it.telecomitalia.centodiciannove_3.apk
- TRANSPORTATION
    - com.navfree.android.OSM.ALL_8675.apk
    - com.softboom.autovelox_4.apk
    - com.softboom.autovelox_6.apk
    - com.viamichelin.android.viamichelinmobile_2201.apk
    - eu.baroncelli.oraritrenitalia_18.apk
    - org.paoloconte.treni_lite_74.apk
- TRAVEL_AND_LOCAL
    - com.google.android.apps.maps_612001402.apk

    - com.google.android.street_18101.apk
    - com.google.earth_53.apk
    - com.google.earth_61.apk
    - com.sygic.aura_36.apk
    - com.waze_1019562.apk
    - org.vernazza.androidfuel_74.apk
- WEATHER
    - com.Meteosolutions.Meteo3b_849.apk
    - com.accuweather.android_25.apk
    - com.gau.go.launcherex.gowidget.weatherwidget_28.apk
    - com.gau.go.launcherex.gowidget.weatherwidget_32.apk
    - com.ilmeteo.android.ilmeteo_18.apk
    - com.ilmeteo.android.ilmeteo_24.apk
    - com.weather.Weather_30802.apk

El listado de aplicaciones malware para analizar ha sido:

- MALWARE
    - Andr_PJApps-Gen_f051eeab57e42d569d298ad076c9fb47610e201e.apk
    - anserverb.apk
    - anserverb_qqgame.apk
    - BlackList_Pro_v2.8.apk_02128bc06f2b442388353879c7188634a83378795d29c5dfc7216bab2e3377cb
    - BlackList_Pro_v2.8.apk_68019a986aee3e698f7d141e7900f4c4f3d444de3ab18aa814e1a424f72ccd07
    - BloodvsZombie_com.gamelio.DrawSlasher_1_1.0.1.apk
    - _com.aijiaoyou.android.sipphone_1005_1.0.5.apk
    - _com.allen.txthej_1_1.0_F438ED38B59F772E03EB2CAB97FC7685.apk
    - com.Beauty.Breast-1.apk
    - com.Beauty.Girl-1.apk
    - com.Beauty.Leg-1.apk
    - com.crazyapps.angry.birds.rio.unlocker-1.apk
    - _com.electricsheep.master.paintpro_10_2.0.1.apk
    - com.keji.sendere_c3b9ed157b71fba7c01be4394c12cd01.apk
    - _com.keji.unclear_1_1.0_BC6C20C79AED279B409C614A92E63BB9.apk
    - _com.mj.iMatch_1_1.0-0e51a56cc59fa3361b48cb9425a03b57.apk
    - _com.RZStudio.game.cube_3_2.3-dbcc8df8cad771ef7bc807764fed06af.apk
    - _com.sansec_9_V1.0.09.apk
    - _com.tutusw.onekeyvpn_7_1.1.6.apk
    - copy9_23.apk
    - de.mehrmannd.sdbooster-GAMEX.apk
    - HotGirls3_com.japanese.hot.girl_1_1.0.apk
    - iCalendaracbcad45094de7e877b656db1c28ada2.apk
    - instagram.apk
    - net.maxicom.android.snake_7937c1ab615de0e71632fe9d59a259cf.apk

- Newfpwap_com_liveprintslivewallpaper.apk
- org.expressme.love.ui.apk
- PhoneLocator_Pro_4.6.apk_ec89d9bcc4fdcd24714fa9c8fd15b2194093cfea35064e1424b60f919ed1955d
- QQ_tencent.qqgame.lord_24_1.1.apk
- Update.apk
- v1.0_com.GoldDream.pg_1_1.0_F66EE5B8625192D0C17C0736D208B0BD.apk
- Zitmo_tr_ECBBCE17053D6EAF9BF9CB7C71D0AF8D.apk

ANEXO 3. Tabla de Resultados del Estudio

Category	Apk	INSERT Posible Vulnerabilities	SELECT Posible Vulnerabilities	UPDATE Posible Vulnerabilities	DELETE Posible Vulnerabilities	Fail Positives	TOTAL	Classes
APP_WALLPAPER	com.badoo.mobile_49.apk	0	2	0	0	0	2	com/facebook/android/FacebookFQL$2
APP_WIDGETS	com.facebook.katana_48315.apk	0	1	0	0	0	1	com/facebook/orca/protocol/methods/FetchAppConfigMethod
APP_WIDGETS	com.facebook.katana_79836.apk	0	1	0	0	0	1	com/facebook/orca/protocol/methods/FetchAppConfigMethod
APP_WIDGETS	com.google.android.apps.plus_311328793.apk	0	0	1	0	0	1	com/google/android/apps/plus/content/EsDatabaseHelper
APP_WIDGETS	com.google.android.gm_176.apk	0	2	1	0	0	3	com/google/android/gm/provider/MailEngine, com/google/android/gm/provider/MailCore
BOOKS_AND_REFERENCE	-	0	0	0	0	0	0	-
BUSINESS	-	0	0	0	0	0	0	-
COMICS	-	0	0	0	0	0	0	-
COMMUNICATION	com.facebook.orca_49466.apk	0	1	0	0	0	1	com/facebook/orca/protocol/methods/FetchAppConfigMethod
COMMUNICATION	com.google.android.gm_176.apk	0	2	1	0	0	3	com/google/android/gm/provider/MailEngine, com/google/android/gm/provider/MailCore
EDUCATION	com.studiociriello.quiz.patente.autofree_10.apk	0	8	0	0	0	8	com/millennialmedia/android/AdDatabaseHelper
ENTERTAINMENT	com.wifi.hacker.cracker_88889.apk	0	10	0	0	0	10	com/millennialmedia/android/AdDatabaseHelper
FINANCE	-	0	0	0	0	0	0	-
GAME	com.halfbrick.fruitninjafree_1603.apk	0	0	0	1	1	0	com/admob/android/ads/AdView
HEALTH_AND_FITNESS	com.macropinch.hydra.android_24.apk	0	8	0	0	0	8	com/millennialmedia/android/AdDatabaseHelper
LIBRARIES_AND_DEMO	-	0	0	0	0	0	0	-
LIFESTYLE	com.ikea.catalogue.android_17.apk	0	4	0	13	0	17	com/ec/hana/core/data/DBSettings, com/ikea/catalogue/android/FreeScrollView, com/ikea/catalogue/android/SearchList, com/ikea/catalogue/android/DeleteQueryCache
MEDIA_AND_VIDEO	-	0	0	0	0	0	0	-
MEDICAL	appinventor.ai_kosma822.IpnosiRegressivaLite_7.apk	0	2	0	0	2	0	com/google/appinventor/components/runtime/PhoneCall, com/google/appinventor/components/runtime/Texting
MEDICAL	mm.app.formulazioni_1.apk	0	2	0	0	0	2	mm/app/formulazioni/principi_attiviFragment, mm/app/formulazioni/tuttifarmaciFragment
MUSIC_AND_AUDIO	com.melodis.midomiMusicIdentifier.freemium_10517.a	0	10	0	0	0	10	com/millennialmedia/android/AdDatabaseHelper

Category	Apk	INSERT Posible Vulnerabilities	SELECT Posible Vulnerabilities	UPDATE Posible Vulnerabilities	DELETE Posible Vulnerabilities	Fail Positives	TOTAL	Classes
	pk							
MUSIC_AND_AUDIO	com.shazam.android_76414.apk	0	1	0	0	0	1	com/shazam/library/LibraryDAO
MUSIC_AND_AUDIO	com.shazam.android_76601.apk	0	1	0	0	0	1	com/shazam/library/LibraryDAO
NEWS_AND_MAGAZINES	com.jappit.android.televideo_6.apk	0	0	0	1	1	0	com/admob/android/ads/AdView
NEWS_AND_MAGAZINES	com.tilab_8.apk	1	0	0	0	0	1	com/tilab/DbAdapter$DatabaseHelper
NEWS_AND_MAGAZINES	it.froggy.tgcom_11.apk	0	3	0	2	0	5	it/froggy/tgcom/TGComDB
NEWS_AND_MAGAZINES	it.froggy.tgcom_13.apk	0	3	0	2	0	5	it/froggy/tgcom/TGComDB
PERSONALIZATION	com.gau.go.launcherex_126.apk	0	0	5	0	0	5	com/jiubang/ggheart/data/DatabaseHelper
PERSONALIZATION	dapks/com.gau.go.launcherex_142.apk	0	0	5	0	0	5	com/jiubang/ggheart/data/DatabaseHelper
PERSONALIZATION	com.srsdev.wallpapers_319.apk	0	10	0	0	0	10	com/millennialmedia/android/AdDatabaseHelper
PHOTOGRAPHY	com.fingersoft.cartooncamera_13.apk	0	1	0	0	0	1	com/getjar/sdk/data/DBAdapterRunningApps
PRODUCTIVITY	-	0	0	0	0	0	0	-
SHOPPING	com.ebay.annunci_15.apk	0	1	2	2	0	5	com/ebay/app/data/workers/CategoryDBWorker, com/ebay/app/data/workers/PostAdAttributeDBWorker
SOCIAL	com.facebook.katana_48315.apk	0	1	0	0	0	1	com/facebook/orca/protocol/methods/FetchAppConfigMethod
SOCIAL	com.facebook.katana_79836.apk	0	1	0	0	0	1	com/facebook/orca/protocol/methods/FetchAppConfigMethod
SOCIAL	com.google.android.apps.plus_311328793.apk	0	0	1	0	0	1	com/google/android/apps/plus/content/EsDatabaseHelper
SPORTS	-	0	0	0	0	0	0	-
TOOLS	com.devuni.flashlight_137.apk	0	10	0	0	0	10	com/millennialmedia/android/AdDatabaseHelper
TOOLS	com.devuni.flashlight_138.apk	0	10	0	0	0	10	com/millennialmedia/android/AdDatabaseHelper
TRANSPORTATION	org.paoloconte.treni_lite_74.apk	0	1	0	0	0	1	org/paoloconte/treni_lite/Utility
TRAVEL_AND_LOCAL	-	0	0	0	0	0	0	-
WEATHER	com.accuweather.android_25.apk	0	10	0	0	0	10	com/millennialmedia/android/AdDatabaseHelper
WEATHER	com.ilmeteo.android.ilmeteo_18.apk	2	10	1	0	0	13	com/ilmeteo/android/ilmeteo/data/Utils, com/ilmeteo/android/ilmeteo/SendActivity$2
WEATHER	com.ilmeteo.android.ilmeteo_24.apk	2	11	1	0	0	14	com/ilmeteo/android/ilmeteo/data/Utils, com/ilmeteo/android/ilmeteo/SendActivity$2
MALWARE	_com.aijiaoyou.android.sipphone_1005_1.0.5.apk	1	1	1	1	0	4	com/aijiaoyou/android/sipphone/HistoryManager

Category	Apk	INSERT Posible Vulnerabilities	SELECT Posible Vulnerabilities	UPDATE Posible Vulnerabilities	DELETE Posible Vulnerabilities	Fail Positives	TOTAL	Classes
MALWARE	com.Beauty.Breast-1.apk	0	0	0	1	1	0	com/admob/android/ads/AdView
MALWARE	com.Beauty.Girl-1.apk	0	0	0	1	1	0	com/admob/android/ads/AdView
MALWARE	dapks/com.Beauty.Leg-1.apk	0	0	0	1	1	0	com/admob/android/ads/AdView
MALWARE	_com.electricsheep.master.paintpro_10_2.0.1.apk	0	0	0	1	1	0	com/admob/android/ads/AdView
MALWARE	_com.mj.iMatch_1_1.0-0e51a56cc59fa3361b48cb9425a03b57.apk	0	0	0	1	1	0	com/admob/android/ads/AdView
MALWARE	iCalendaracbcad45094de7e877b656db1c28ada2.apk	0	0	0	1	1	0	com/admob/android/ads/AdView
TOTALS		**6**	**128**	**19**	**28**	**10**	**171**	

Tabla 86. Tabla de Resultados del Estudio

Bibliografía

ALI perfiles. (2011). *www.ali.es.* Recuperado el 14 de Junio de 2014, de www.ali.es: http://www.ali.es/modules/miprofesion/item.php?itemid=35

Apk Files. (2014). *developer.android.com.* Recuperado el 9 de Junio de 2014, de developer.android.com: http://developer.android.com/tools/building/index.html

Apktool. (2014). *https://code.google.com/p/android-apktool/.* Recuperado el 6 de Junio de 2014, de https://code.google.com/p/android-apktool/: https://code.google.com/p/android-apktool/

Binary file. (2014). *en.wikipedia.org.* Recuperado el 12 de Junio de 2014, de en.wikipedia.org: http://en.wikipedia.org/wiki/Binary_file

Denney, R. (2003). *www.agileconnection.com.* Recuperado el 15 de Junio de 2014, de www.agileconnection.com: http://www.agileconnection.com/article/calculating-roi-your-investment-requirements-management-tools

DynaTAC 8000x. (2014). *www.flickr.com.* Recuperado el 2 de Junio de 2014, de www.flickr.com: https://www.flickr.com/photos/arteaoavesso/3479764727/sizes/m/in/photostream/

Ejemplo iGoat. (2014). *www.owasp.org.* Recuperado el 6 de Junio de 2014, de www.owasp.org: https://www.owasp.org/index.php/Mobile_Top_10_2012-M1_Insecure_Data_Storage

exeSQL in Android Developer Web. (2014). *developer.android.com.* Recuperado el 16 de Junio de 2014, de developer.android.com: http://developer.android.com/reference/android/database/sqlite/SQLiteDatabase.html#execSQL(java.lang.String)

Forbes. (2013). *www.forbes.com.* Recuperado el 12 de Junio de 2014, de www.forbes.com: http://www.forbes.com/sites/louiscolumbus/2013/01/17/2013-roundup-of-mobility-forecasts-and-market-estimates/

Fragments. (2014). *developer.android.com.* Recuperado el 9 de Junio de 2014, de developer.android.com: http://developer.android.com/reference/android/app/Fragment.html

GNU GPL. (2007). *www.gnu.org.* Recuperado el 15 de Junio de 2014, de www.gnu.org: http://www.gnu.org/copyleft/gpl.html

KeyChain in Android Developer Web. (2014). *developer.android.com.* Recuperado el 16 de Junio de 2014, de developer.android.com: http://developer.android.com/reference/android/security/KeyChain.html

Khurshid, K. (2013). Comparison survey of 4G competitors (OFDMA, MC CDMA, UWB, IDMA). *Aerospace Science & Engineering (ICASE), 2013 International Conference on* .

LODP Informática. (2014). *www.colordeu.es.* Recuperado el 9 de Junio de 2014, de www.colordeu.es: http://www.colordeu.es/BLOG/lopd-ley-de-proteccion-de-datos-en-tus-aplicaciones-web

LOPD. (1999). *www.boe.es.* Recuperado el 9 de Junio de 2014, de www.boe.es: http://www.boe.es/buscar/doc.php?id=BOE-A-1999-23750

Mobile Phone, E. (2014). *loling1.blogspot.com.es.* Recuperado el 2 de Junio de 2014, de loling1.blogspot.com.es: http://loling1.blogspot.com.es/2013/06/evolution-of-mobile-phone.html

Modelo de Datos Android. (2014). *abdennour-insat.blogspot.com.es.* Recuperado el 5 de Junio de 2014, de abdennour-insat.blogspot.com.es: http://abdennour-insat.blogspot.com.es/2012/04/summary-of-matoss-android-tutos.html

MVC Android 2. (2014). *gkonandroid.blogspot.com.es.* Recuperado el 10 de Junio de 2014, de gkonandroid.blogspot.com.es: http://gkonandroid.blogspot.com.es/2013/11/mvc-architecture-in-android.html

MVC Android. (2014). *androideity.com.* Recuperado el 10 de Junio de 2014, de androideity.com: http://androideity.com/2012/05/10/la-importancia-del-mvc-en-android/

Netsparker. (2014). *www.netsparker.com.* Recuperado el 5 de Junio de 2014, de www.netsparker.com: https://www.netsparker.com/

OWASP DroidGoat. (2014). *www.owasp.org.* Recuperado el 6 de Junio de 2014, de www.owasp.org: https://www.owasp.org/index.php/Projects/OWASP_GoatDroid_Project#tab=Main

OWASP iGoat. (2014). *www.owasp.org.* Recuperado el 5 de Junio de 2014, de www.owasp.org: https://www.owasp.org/index.php/OWASP_iGoat_Project#tab=Main

OWASP Top 10 Mobile Risks. (2014). *www.owasp.org.* Recuperado el 6 de Junio de 2014, de www.owasp.org: https://www.owasp.org/index.php/OWASP_Mobile_Security_Project#tab=Top_10_Mobile_Risks

OWASP WebGoat. (2013). *www.owasp.org.* Recuperado el 5 de Junio de 2014, de www.owasp.org: https://www.owasp.org/index.php/Category:OWASP_WebGoat_Project

OWASP. (2014). *www.owasp.org.* Recuperado el 5 de Junio de 2014, de www.owasp.org: https://www.owasp.org/index.php/About_OWASP

PBE in Android Developer Web. (2014). *developer.android.com.* Recuperado el 16 de Junio de 2014, de developer.android.com: http://developer.android.com/reference/javax/crypto/spec/PBEKeySpec.html

Pentesting. (2013). *www.forbes.com.* Recuperado el 6 de Junio de 2014, de www.forbes.com: http://www.forbes.com/sites/ericbasu/2013/10/13/what-is-a-penetration-test-and-why-would-i-need-one-for-my-company/

rawQuery in Android Developer Web. (2014). *developer.android.com.* Recuperado el 13 de Junio de 2014, de developer.android.com: http://developer.android.com/reference/android/database/sqlite/SQLiteDatabase.html#rawQuery(java.lang.String, java.lang.String[])

Regular expressions in grep. (2010). *www.cyberciti.biz.* Recuperado el 16 de Junio de 2014, de www.cyberciti.biz: http://www.cyberciti.biz/faq/grep-regular-expressions/

SQLite features. (2014). *www.sqlite.org.* Recuperado el 5 de Junio de 2014, de www.sqlite.org: http://www.sqlite.org/features.html

SQLite Injection. (2014). *www.tutorialspoint.com.* Recuperado el 2 de Junio de 2014, de www.tutorialspoint.com: http://www.tutorialspoint.com/sqlite/sqlite_injection.htm

SQLite Insert in Android Developer Web. (2014). *developer.android.com.* Recuperado el 13 de Junio de 2014, de developer.android.com: http://developer.android.com/reference/android/database/sqlite/SQLiteDatabase.html#insert(java.lang.String, java.lang.String, android.content.ContentValues)

Unix ls. (2014). *www.unix.com.* Recuperado el 12 de Junio de 2014, de www.unix.com: http://www.unix.com/man-page/opensolaris/1/ls/

Unix wc. (2014). *www.unix.com.* Recuperado el 2014, de www.unix.com: http://www.unix.com/man-page/opensolaris/1/wc/

WebCruiser. (2014). *sec4app.com.* Recuperado el 5 de Junio de 2014, de sec4app.com: http://sec4app.com/

Yeh, C.-H. (2008). Design and implementation of honeypot systems based on open-source software. *Intelligence and Security Informatics, 2008. ISI 2008. IEEE International Conference on* .